ISSN 2782-2427

CONTROL SCIENCES

2/2025

[1]Russian Academy of Sciences.

CONTROL SCIENCES
Scientific Technical Journal

6 issues per year
ISSN 2782-2427
Open access

Published since 2021

Original Russian Edition
Problemy Upravleniya
Published since 2003

FOUNDER AND PUBLISHER
V.A. Trapeznikov
Institute of Control Sciences
of Russian Academy of Sciences

Editor-in-Chief
D.A. Novikov, RAS Academician

Deputy Editor-in-Chief
F.F. Pashchenko

Executive Editor-in-Chief
N.E. Maximova

Editor
L.V. Petrakova

Editorial address
65 Profsoyuznaya st., office 410,
Moscow 117997, Russia

☎/🖹 +7(495) 198-17-20, ext. 1410

✉ pu@ipu.ru

URL: http://controlsciences.org

Published: June 16, 2025

Registration certificate of
Эл № ФС 77-80482
of 17 February 2021
issued by the Federal Service
for Supervision of Communications,
Information Technology, and Mass
Media

CONTROL SCIENCES

2.2025

CONTENTS

TECHNICAL CONDITION MONITORING METHODS TO MANAGE THE REDUNDANCY OF SYSTEMS. PART I: Built-in Control and Partition into Classes

V. N. Bukov*, A. M. Bronnikov**, A. S. Popov***, and V. A. Shurman****

*Airborne Aeronautical Systems, Moscow, Russia
**Bauman Moscow State Technical University, Moscow, Russia
***Zhukovsky–Gagarin Air Force Academy, Voronezh, Russia
****Institute of Aircraft Equipment, Zhukovsky, Russia

*✉ v_bukov@mail.ru, **✉ bronnikov_a_m@mail.ru, ***✉ saga30@yandex.ru, ****✉ shurman@niiao.ru

Abstract. Redundancy management of a technical system involves a monitoring procedure (control of the current state of its components) to reconfigure the system and improve the performance and autonomy of its application. This paper initiates a four-part survey of the state-of-the-art monitoring methods for redundancy management. Part I is mainly devoted to the analysis of voting schemes, fidelity rules, control codes, and program control, representing the most widespread monitoring methods in modern technical systems and built-in control. In addition, we examine long-known, albeit less common, monitoring methods: diagnosis with partition into classes and diagnosis based on algebraic invariants.

Keywords: technical condition monitoring, redundancy management, diagnosis, built-in control, control codes, partition into classes, algebraic invariants.

INTRODUCTION

The increased capabilities of information and mathematical support of control processes in complex dynamic systems and equipment complexes enable a fundamentally new approach to meeting the ever-tightening requirements for their fault tolerance, particularly based on manageable redundancy [1, 2]. One task of redundancy management in advanced systems is to perform a monitoring procedure, i.e., track current changes [3] in the operational readiness of the components of such systems [4, 5] in order to reconfigure them whenever necessary.

Technical condition monitoring consists in observing the state of a given object during an interval of its life cycle (e.g., an aircraft flight). This process is based on a certain hierarchy of methods for determining the technical condition of an object, i.e., technical diagnosis [6]. According to [7], monitoring is an integral part of maintenance.

Technical diagnosis is usually a discrete sequence of technical diagnoses (diagnostic results bound to certain time instants) [6].

Diagnosing the technical condition of technical systems is a very complex problem requiring a wide range of algorithmic solutions. From an engineering point of view, the content of control (diagnosis) of systems is to detect (find) faults by available features.

Concerning the prospects of this R&D direction, the strongest results in the field of monitoring should be expected from a system approach[1], primarily based on the triunity of the following key directions: 1) reliability and operability, 2) rationality (limitation of resources used, essential, e.g., for self-checking circuits [8]), and 3) the reasonable depth (“granularity”) of the

[1] This idea was suggested by one of the paper’s reviewers, and we express sincere gratitude for it. However, the scientific substantiation and revelation of the corresponding considerations is the subject of a separate publication.

system design, optimizing the balance of large- and small-fragment partitions into system components.

The figure shows the classification of currently used and being developed diagnosis methods for dynamic systems, with keywords characterizing them rather than their conventional names. On the one hand, test and functional control have much in common in the methods and procedures used and, on the other, they possess some peculiarities not discussed in the survey.

Below we summarize the main approaches to diagnosing the faults of components of aircraft on-board equipment complexes (OECs). This survey does not claim to be exhaustive. We endeavor to treat the subject systematically and illustrate the capabilities and typical limitations of common approaches as well as their development trends. The presentation evolves from the simplest (obvious) methods to more and more complex ones, requiring the developer's knowledge of special mathematical apparatus.

1. MONITORING BASED ON BUILT-IN CONTROL

Built-in control (BiC) [9, 10] is among the most widespread modern solutions for ensuring the required fault tolerance of various technical systems. In systems of increased danger (particularly aircraft OECs), built-in control is implemented for all components (systems, subsystems, assemblies, modules, and even microcircuits).

BiC is a set of hardware or software components, introduced into systems, their parts, or functional assemblies (FAs). As a rule, they do not participate in the work of functional modules (FMs) of the system or its FAs on purpose but collect and summarize various data that objectively reflect the operability of these modules in the developer's opinion. Looking forward, BiC can be based on various monitoring methods and their combinations, covered in all four parts of the survey. Here we consider only the most widespread approaches, which are widely implemented in modern BiC.

There are two different organizational approaches to the operation of BiC: test control and functional control.

1.1. Test Control

In test control [11, 12], assemblies, devices, and the entire system are checked using special equipment, namely, generators of test (input) impacts and analyzers of output responses. Due to the need for additional

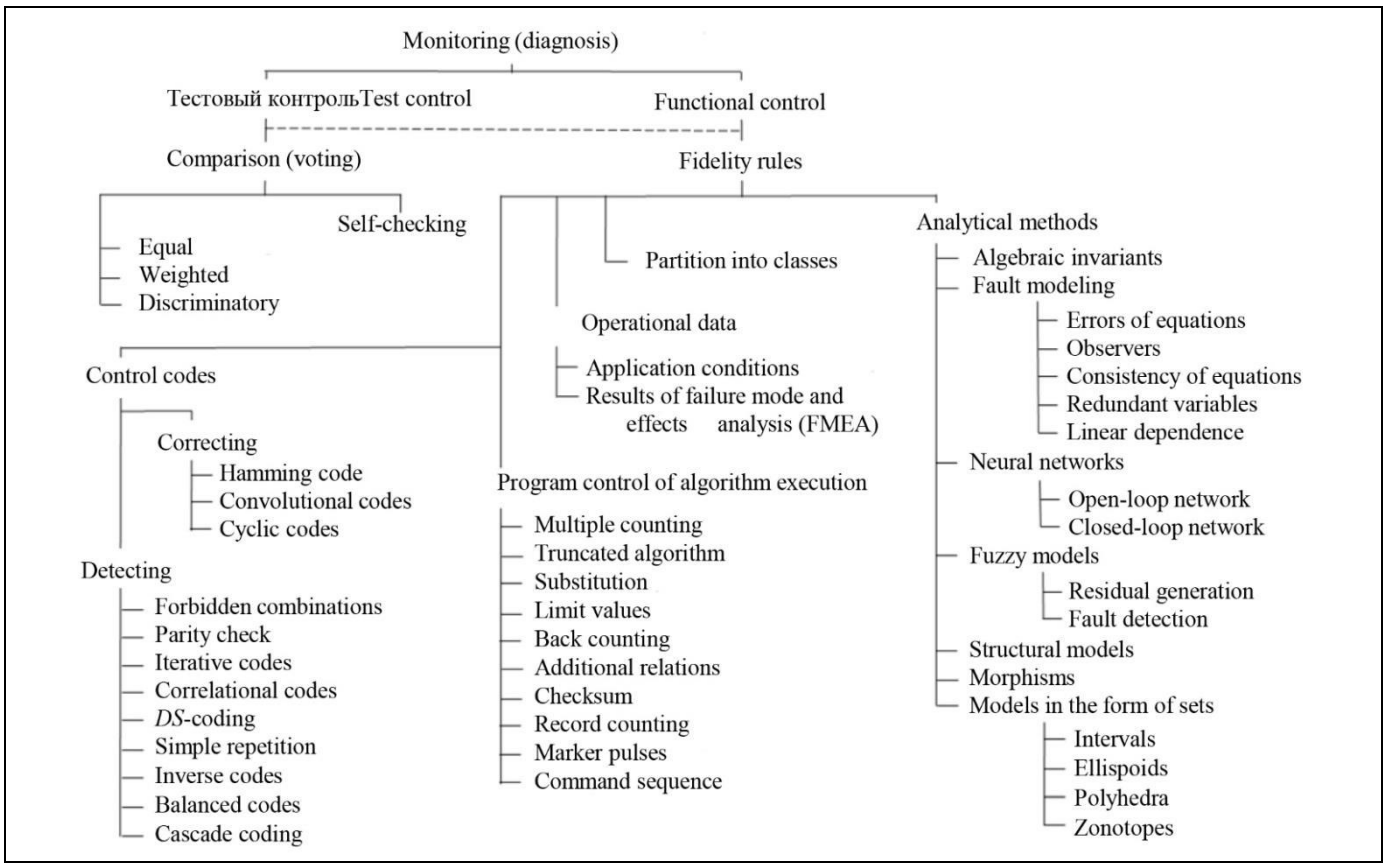

Fig. Diagnosis methods for dynamic systems: a classification.

equipment and the complexity of combining with normal operation, test methods are applied only when the controlled object is not used for its intended purpose.

In onboard conditions, testing is performed using specialized BiC tests in a background mode during special intervals (slots) allocated by the real-time operating system. The content of BiC tests is the comparison of the results of addressing (writing and reading) software-accessible resources of the computing unit, including specially organized control channels.

Testing with simulation of standard impacts is performed by a special generator of tests, and the output responses are compared with the reference ones using an analyzer.

Probabilistic testing involves a test generator of pseudorandom impacts and statistical processing of the output data; the results are compared with the reference ones obtained beforehand.

Testing with switch counting includes generating a sequence of test sets of signals at the circuit's input and calculating the number of switches at its output; the result is compared with the reference.

In *signature testing,* the controlled object is stimulated using a generator of pseudorandom impacts, and the output responses are compressed through a signature analyzer; the resulting signatures are compared with the reference signatures.

1.2. Functional Control

Functional control is performed during the intended-purpose operation of the controlled object and is generally implemented based on two main principles: voting schemes and fidelity rules.

The principal peculiarity of *comparison (voting) schemes* is the simultaneous use of several technical devices (subsystems, assemblies, or modules) identical in purpose and implementation. The diagnosis system is reduced to the means of comparing the data of these systems and selecting a preference by a given rule of comparing their outputs. Here, a common solution is the so-called quorum elements (QEs), which identify faulty modules by processing the voting results of several connected FMs. The operability of an FM is judged by a significant deviation of its output from those of same-type modules (the largest deviation or that exceeding a given threshold) [13–15].

The main peculiarities of the quorum-based method include:

– the assumption that the technical state of an FM remains unchanged within a cycle;

– the assumption that a QE is operable (never fails)[2];

– applicability to three or more FMs (in the case of two FMs, a pair of FMs becomes the controlled object, not each FM separately);

– the assumption that within the voting rules (equal, weighted, with discriminations, etc.), the operable FMs within each cycle dominate the faulty ones and the latter can be disconnected;

– a common data flow for all FMs.

A peculiar form of comparison is widely implemented in the so-called self-checking systems [8]: a set of same-type modules subjected to identical input actions is divided into pairs, and the outputs within each pair are compared with each other. A pair with matching outputs is considered to be operable; otherwise, both modules of the pair are considered to be inoperable.

The principal peculiarity of systems using *fidelity rules* (FRs) is the presence of a single diagnosed device. Depending on particular conditions and solutions, such rules can be as follows: comparing with reference (electronic) models, detecting violations of given time and (or) parametric intervals (control by parameter tolerance [13]), checking logical and other relations, calculating different-order invariants, etc.

The main peculiarities of the method of FRs include the following:

– Within each cycle, the operability of an FM does not change.

– By assumption, an element implementing FRs is operable. (If there is a reference model, it is operable.)

– This method is applicable to any number of FMs.

– By assumption, the input and output data contain sufficient information.

– Each FM has a separate data flow.

The recent direction [16] stands somewhat apart. It can be called FM monitoring based on operational data with recording of application conditions. By assumption, a special element (chip) is structurally and functionally connected directly to an FM to gather and accumulate data on the conditions of its use and storage. Such a chip stores different parameters (FM data) and sends them to the monitoring module, in particular:

– passport information,

– test results at different stages of the life cycle,

[2] The matter concerns a conceptual solution; however, multilevel majorization schemes are known that shift this constraint to higher levels of comparison of the results.

– statistics of operation indicators and characteristics (estimates of the achieved accuracy, remaining life, energy indicators, etc.),

– statistics of external impacts during intended use, storage, and routine maintenance.

The monitoring module is responsible for analyzing the incoming data and judging FM operability based on the analysis results.

1.3. Advantages and Conditions of Using Built-in Control

Thus, we summarize the common peculiarities (limitations) of BiC with different degrees of occurrence:

– weak[3] assumptions about the unchanged operability of the controlled devices within the monitoring cycle;

– strong[4] assumptions about the operability of control systems or their major devices;

– the requirement for a minimum admissible or large number of FMs (in the case of quorum or majority control);

– the requirement that operable FMs dominate inoperable FMs;

– the fast disconnection of faulty FMs;

– the sufficient informativeness requirement for all processes in FMs.[5]

The main advantage of using BiC (in the current form) to monitor the components of a redundant OEC is the well-established technologies of their creation and application in practice.

2. USE OF CONTROL CODES

A specific direction of built-in control of digital devices is the use of control codes to detect and correct errors in digital data [11, 17–23].

Block codes are most widespread: a symbolic sequence at the source output is divided into blocks (codewords, or code combinations) containing the same number of symbols. Any code can detect and correct errors (is noise-resistant) if some of its codewords are not used for information transmission [24]. In other words, a noise-resistant code must be redundant. Nevertheless, two types of noise-resistant codes are distinguished: codes with error detection and codes with error correction (correcting codes).

Error detection consists in identifying the transformation of the received or read (allowed) codeword into the so-called forbidden one. Note that the errors related to its transformation into another authorized codeword are not detected.

Error correction is a more complex operation: all forbidden codewords are divided into disjoint subsets, and each subset is assigned to one of the allowed codewords. Thus, the belonging of the resulting forbidden codeword to a subset is interpreted as the corresponding allowed codeword. If the resulting forbidden codeword belongs to neither of the subsets, the error will be detected but not corrected.

The error detection and correction properties of codes are characterized by the detection (K_{det}) and correction (K_{cor}) coefficients, $K_{\text{det}} > K_{\text{cor}}$, which have a probabilistic nature.

Detecting codes include, e.g., the following common codes.

A forbidden combination check code detects combinations of bit values in a codeword that are declared invalid (e.g., accessing a non-existent address).

A parity check code can be treated as a special case of a forbidden combination check code. It is formed by adding one non-informative bit to the information bits storing a codeword (mod2 convolution, supplementing the number of units in a code to oddness, checked at each exchange between registers). A parity check code is simple in technical realization and detects errors of odd multiplicity.

Iterative codes belong to the class of *product codes* and can be written as rectangular matrices or tables (can be built from matrices of higher dimensions). The information symbols recorded by rows and columns can be encoded by a noise-resistant code of the same or different types.

In a particular case, iterative codes are an evolution of parity check codes: they are used for the separate parity checking of rows, columns, and other structures of stored and transmitted data arrays. Such codes are characterized by simplicity and efficiency in detecting multiple errors. As an illustration, we present the control principle of a two-dimensional matrix array with parity checking by rows, columns, and the principal diagonal [22]:

1	0	1	1	1	$\text{par}_{1j}=0$
0	0	1	0	1	$\text{par}_{2j}=0$
1	1	1	0	0	$\text{par}_{3j}=1$
0	1	0	0	1	$\text{par}_{4j}=0$
1	1	1	0	1	$\text{par}_{5j}=0$
$\text{par}_{i1}=1$	$\text{par}_{i2}=1$	$\text{par}_{i3}=0$	$\text{par}_{i4}=1$	$\text{par}_{i5}=0$	$\text{par}_{ii}=1$

.

[3] This assumption is not crucial in practice.

[4] This assumption significantly narrows the applicability of the approach.

[5] The need for this requirement will be illustrated in part III of the survey.

Correlational codes involve an additional control bit introduced for each information bit of a word so that the entry "01" corresponds to the initial information value "0" and the entry "10" to the value "1." In this case, the codes "00" and "11" are signs of data distortion.

DS-coding [23] is a kind of using correlation codes. It is implemented via two parallel channels of sequential code transmission. If there is a bit value change from 0 to 1 or from 1 to 0 in the main channel *D*, no bit value change will occur in the control channel *S*. And vice versa, if there is no bit value change in the channel *D*, the bit value will change from 0 to 1 or from 1 to 0 in the channel *S*. The receiver controls the absence of either a simultaneous change or constancy of bit values in both channels. If this condition is violated, a data transmission error will be detected. This type of coding is characterized by a minimum error detection delay (one stroke).

A simple repetition code involves the repeated transmission of codewords. If they coincide, then the absence of an error is confirmed; otherwise, errors are detected.

An inverse code is a modification of a simple repetition code: in the case of an odd number of units in a source word, its inversion is added to it. The inverse code is received in two stages. In the first stage, the units in the base word are summed. If the number of units is even, the control bits will be received without change; if odd, the control bits will be inverted. In the second stage, the control and information bits are summed modulo 2. The zero sum indicates the absence of errors. If the sum is non-zero, the received word will be rejected.

Balanced codes are the simplest block codes in which allowed words contain a fixed number of units. They are used mainly for data transmission via communication channels.

Cascade coding. The advantages of different coding methods can be combined by applying cascade coding. In this case, information is first encoded with one code and then with another, resulting in a *product code*.

Correcting codes include, but are not limited to, the following.

A Hamming code is one of the most widely known classes of linear codes [17, 20, 24, 25]. In this code, the bits with numbers representing the degree of two are control bits, and the rest are information bits. As a rule, the maximum possible number of information bits is determined based on the number of control bits. There are no Hamming codes with one control bit; a Hamming code with two control bits contains one information bit, etc. The result is achieved by repeatedly checking the received combination for parity. Each check must cover part of the information bits and one of the redundant bits. When transmitting data (writing data to memory), the values of the control bits are formed and written; when receiving (reading) these data, the control bits are again formed and compared with the original ones. If all the newly calculated control bits coincide with the received ones, then the message contains no errors; otherwise, an error is detected and, if possible, this error is corrected.

In *convolutional codes* [26], control bits are formed based on several information words and decoding is performed based on several codewords. The principle of convolutional codes can be compared with error correction by sense. Convolutional codes are also called lattice or trefoil codes.

In *cyclic codes* [17–19], a cyclic permutation of any codeword containing both information and control bits results in a word belonging to the same cyclic code. The formalism of operations over polynomials is used to construct cyclic codes: polynomials corresponding to the received codewords must divide by their generator without a remainder. The presence of a remainder indicates an error. If the number of errors does not exceed the calculated value, then the remainder depends on the configuration of errors and can be used for their correction. Cyclic codes allow simplifying the circuit implementation of encoding and decoding devices by using shift registers.

3. PROGRAM METHODS TO CONTROL ALGORITHM EXECUTION

The advantages of software tools include versatility, flexibility, and relatively low cost. At the same time, they require specific software packages for implementation.

The *multiple counting* method assumes that the control task is solved two or more times. In the simplest case, the coinciding results are a sign of the correct solution. A deeper approach may involve various algorithms to process the results, including voting schemes (majorized estimates). Additional memory and counting time are required.

Control by *the truncated algorithm* is intended to reduce the cost of multiple counting. It applies to the cases when the task has a simplified (reduced) algorithmic variant, and therefore a less accurate but substantially similar result can be formed. Such a variant may not satisfy the customer as the solution of the original task, but it is acceptable for assessing the correctness of the full algorithm.

The limit value check method.[6] It can be used in problems with a priori estimation of admissible ranges of the solution. Often such ranges are determined for separate checkpoints (places in the action sequence of an algorithm). This method can be treated as a variant of the truncated algorithm when it is applied to calculate the bounds of possible solutions.

The substitution method. If the algorithm to be checked solves mathematical equations, then traditionally an effective check is to substitute the resulting solution into the original equations. An admissibly small residual (the difference between the left- and right-hand sides) of the equations allows judging the correct solution. Unlike multiple counting, substitution reveals systematic (programming) errors. In addition, substitution is usually less labor-intensive than multiple counting.

The back-counting method. In some tasks, it is possible to determine the initial data by the result obtained, and the correctness of counting can be checked accordingly. The corresponding costs can sometimes be smaller than those of direct counting.

Checking by *additional relations*. In this method, it is possible to introduce relations between various parameters of the main problem, described by exact or statistical formulas. Generally speaking, it is a simplified version of the analytical methods: the method of invariants and the method of redundant variables (see part II of the survey).

The *checksum method* consists in summing up all words of an array (commands, data) and then saving the sum in a definite part of the array. In the interest of control, repeated summation and comparison with the checksum saved are performed. It is applied mainly when transferring data and uploading/downloading programs.

The record counting method is to calculate and memorize the records being executed, i.e., the datasets precisely defined. Later, when handling the data, the counting is repeated and the result is compared with the original one. This method detects losses or omissions in data processing.

Marker pulse control allows tracking the passage of certain positions by a computational process or the completion of counting. The generation of appropriate time points (markers) must be provided in the algorithm being implemented. In case of violating the prescribed marker sequence or exceeding the waiting time, the counting is interrupted and a decision on further actions is made (recalculation, use of reserve variants, or task stop).

A specific case of the marker pulse method is the multiple (three to five times) sending of *a data retrieval request*. An error is fixed under no response to all the requests sent. In case of receiving a response to any request, the data transfer[7] process is considered to be fault-free.

Control of the execution sequence of commands and program modules is carried out by dividing programs into sections. Then one of the following methods is applied:

- For each section, the convolution is calculated (by counting the number of operators, by signature analysis, by using codes) and then compared with the pre-calculated value.
- Each section is assigned a certain codeword (section key), which is written to the selected RAM cell before the section execution starts and is checked at the end of this section. The nodes of branching programs are checked using keys and cyclic sections are checked by the number of cycle repetitions.

Unlike these heuristic methods, the diagnosis methods described below proceed from a relatively deeper mathematical analysis of the system diagnosed.

4. DIAGNOSIS WITH PARTITION INTO CLASSES

The mathematical formulation of the control problem was given in [27], where faults were searched in an electrical circuit. The main element of this formulation is a rectangular table of faults containing S feature rows and D state columns. Consider a fixed subset R_D of columns. If R_D is a partition of the set of state-columns into classes[8], then formally the problem is to determine a partition R_S on the set of feature rows to obtain a bijective mapping

$$R_D \Leftrightarrow R_S . \qquad (1)$$

Condition (1) is intuitively clear: on elements from the set D as a test, one constructs at least one element from R_D and assigns to it, by the if-and-only-if implication, an element from R_S.

Various modifications of this approach have become widespread [28].

The methods of this approach involve the model of a diagnosed object described by the table of fault functions R_j^i (see the general form below). Here, D denotes the set of technical states of the object; $d_0 \in D$

[6] A kind of parameter tolerance control [12].

[7] Data fidelity is controlled separately.

[8] By definition, classes are either disjoint or completely coincident.

is its fault-free (operable) state; $d_i \in D$, $i = \overline{1, n}$, are faulty (inoperable) states. Each inoperable state corresponds to a certain fault (failure, defect) $s_i \in S$ and conversely.

Fault functions

R		D				
		d_0	⋯	d_i	⋯	d_n
S	s_1	R_1^0	...	R_1^i	...	R_1^n
	⋮	...	...	...	...	...
	s_j	R_j^0	...	R_j^i	...	R_j^n
	⋮	...	...	...	...	...
	s_k	R_k^0	...	R_k^i	...	R_k^n

In this table, S is the set of features $s_j \in S$, $j = \overline{1,k}$; R_j^i is a fault function compactly written by the formula

$$R_j^i = \Psi^i(s_j). \qquad (2)$$

It represents the system behavior in an analytical, graphical, tabular, or other form. A more detailed description includes input and output signals, lists of elementary checks, and initial conditions (for dynamic objects) [28].

The general formula (2) can be further specified in the light of achieving the desired depth of diagnosis. Often the explicit formula (2) is adopted only for the fault-free state R_j^0, and the faulty states are described with respect to this state.

By assumption, all faults possess detectability and distinguishability: all faults can be unambiguously identified and separated from other faults via some set Π of elementary checks. In this case, by a simple enumeration of elementary checks $\pi_k \in \Pi$, one can partition the fault table into disjoint subsets D_υ, $\upsilon = \overline{1, \lambda}$:

$$\bigcup_{\upsilon=1}^{\lambda} D_\upsilon = D, \;\; D_\upsilon \cap D_\mu = \varnothing, \;\; \upsilon \neq \mu. \qquad (3)$$

Fault tables are used to build both diagnostic algorithms and a physical model of the object that implements diagnostic schemes.

For the same table of fault functions and a given partition of the set D into subsets D_υ, it is generally possible to construct several complete non-redundant tests $T \subset \Pi$ (sets of elementary checks π). The content of many solutions of this approach is to minimize the number of elementary checks. Various tools are used for this purpose: dividing diagnosis tasks into direct and inverse (determining the technical state d_i via a given elementary check π_k and determining the set of checks $\{\pi_k\}$ that distinguish a given pair of faults d_m and d_n, respectively), constructing fault trees, splitting inputs and outputs, etc.

5. DIAGNOSIS BASED ON ALGEBRAIC INVARIANTS

Control with calculating algebraic invariants [28] belongs to analytical methods due to using analytical information (mathematical description) about the operation of the controlled object. It consists in checking some algebraic relations (control conditions) for the set of object's output signals, supplemented (if necessary) by one or more redundant signals. The invariance of control conditions is that, in the absence of faults, they must hold for any input signals and at any time instant.

The operation of a diagnostic device can be briefly described as follows. The device receives input U and output Y signals of the object under check. Based on these signals, auxiliary signals Z are generated to satisfy, together with the signals Y, an algebraic equation resolved with respect to the variable Δ (called the syndrome):

$$\Delta = \Phi(Y, Z) = 0. \qquad (4)$$

The syndrome is invariant with respect to the vector of input signals U. If condition (4) is violated ($\Delta \neq 0$), then the occurrence of a fault is judged. In practice, due to the presence of admissible errors in measurements and calculations, the control condition (4) holds approximately, and diagnosis is performed according to the inequality $|\Delta| \leq \varepsilon$, where ε specifies the output signal tolerance for the diagnostic device. Objects possessing such algebraic invariants are called objects with natural redundancy. Examples are objects that must move along a certain (in particular, phase) trajectory (on a sphere, in a plane, etc.) in a fault-free state. The advantage of such systems is the minimum necessary a priori information about the object and additional diagnostic means. The majority of fault detection schemes have artificially created redundancy.

Now we analyze the sensitivity of the syndrome (4) to faults. Let the syndrome be an m-dimensional vector $\Delta = [\Delta_1 \;\; \cdots \;\; \Delta_m]^{\mathrm{T}}$, and let the possible faults of the object be formalized by an n-dimensional vec-

tor $F = [f_1 \ \cdots \ f_n]^{\mathrm{T}}$. In this case, equation (4) is written as

$$\Delta = \Phi(Y, Z, F) = 0. \qquad (5)$$

Obviously, for the syndrome Δ to respond to a single fault $f_i \neq 0$, it suffices to satisfy the nonzero sensitivity condition

$$\exists j : \frac{\partial \Phi_j}{\partial f_i} \neq 0,$$

which is ensured by design solutions. In the case of a multiple fault (when several faults f_i occur simultaneously), the additional condition

$$\begin{bmatrix} \partial\Phi_1 / \partial f_1 & \cdots & \partial\Phi_1 / \partial f_n \\ \vdots & \ddots & \vdots \\ \partial\Phi_m / \partial f_1 & \cdots & \partial\Phi_m / \partial f_n \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \neq 0$$

(no mutual compensation (5) for the effects of these faults) must be valid. Therefore, under $m = n$ and the maximum rank of the Jacobi matrix, all simultaneous faults f_i are detected. If the number m of the syndrome components Δ_j is smaller than the number n of faults f_i, then the mutual compensation of these effects is possible and, consequently, they will be omitted.

CONCLUSIONS

A classification of diagnosis methods has been proposed, and the content and peculiarities of built-in control have been described. Engineering heuristic monitoring methods have been considered. Monitoring methods based on voting schemes, fidelity rules, control codes, functional control software, fault tables, and algebraic invariants have been briefly characterized. Part II of the survey will deal with diagnosis methods based on classical fault modeling of the diagnosed system. In part III, we will analyze diagnosis methods based on neural networks, fuzzy and structural models, and models in the form of sets. Finally, part IV will be devoted to new approaches to technical diagnosis and combinations of different models and methods.

REFERENCES

1. Bukov, V.N., Bronnikov, A.M., Ageev, A.M., et al., The Concept of Controlled Redundancy of On-Board Equipment Complexes, *Trudy 16-oi Vserossisikoi nauchno-prakticheskoi konferentsii "Nauchnye chteniya po aviatsii, posvyashchennye pamyati N.E. Zhukovskogo"* (Proceedings of the 16th All-Russian Scientific and Practical Conference "Scientific Readings on Aviation, Dedicated to the Memory of N.E. Zhukovsky"), Moscow, 2019, pp. 17–33. (In Russian.)
2. Bukov, V.N., Ageev, A.M., Evgenov, A.V., and Shurman, V.A., *Upravlenie izbytochnost'yu tekhnicheskikh sistem. Supervizornyi sposob upravleniya konfiguratsiyami* (Redundancy Management of Technical Systems. The Supervision Method of Configuration Management), Moscow: INFRA-M, 2023. (In Russian.)
3. *The Great Russian Encyclopedia 2004–2017*. URL: https://old.bigenc.ru/economics/text/2227291?ysclid=lt6tp1tio4782244915 (Accessed February 29, 2024; in Russian.)
4. Pouliezos, A.D. and Stavrakakis, G.S., *Real Time Fault Monitoring of Industrial Processes*, Dordrecht: Kluwer Acad. Publishers, 1994.
5. *DO-297. Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations*. Washington: RTCA Inc., 2005.
6. Naumenko, A.P., *Teoriya i metody monitoringa i diagnostiki: Materialy lektsii* (Theory and Methods of Monitoring and Diagnosis: Lectures), Omsk: Omsk State Technical University, 2017. (In Russian.)
7. *GOST* (State Standard) *R 27.605–2013: Reliability in Engineering. Equipment Maintainability. Diagnostic Check*, Moscow: Standartinform, 2014. (In Russian.)
8. Sogomonyan, E.S. and Slabakov, E.V., *Samoproveryaemye ustroistva i otkazoustoichivye sistemy* (Self-checking Devices and Fault-Tolerant Systems), Moscow: Radio i Svyaz', 1989. (In Russian.)
9. *GOST* (State Standard) *R 56079-2014: Aircraft Items. Flight Safety, Reliability, Testability and Maintainability Indices*, Moscow: Standartinform, 2014. (In Russian.)
10. Dolbnya, N.A., Internal Controls Board Computer System Running Real Time Operation System as an Iterative Aggregate, *Vestnik of Samara State Aerospace University*, 2012, no. 5 (36), pp. 224–228. (In Russian.)
11. Arikan, E., Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels, *IEEE Transactions on Information Theory*, 2009, vol. 55, no. 7, pp. 3051–3073.
12. Bugaeva, A.A. and Denisenko, V.V., Testing Process, Methods and Types of Software Testing, *Sinergiya Nauk*, 2022, no. 72, pp. 92–102. (In Russian.)
13. *Diagnostirovanie i prognozirovanie tekhnicheskogo sostoyaniya aviatsionnogo oborudovaniya* (Technical Condition Diagnosis and Prediction for Aircraft Equipment), Sindeev, I.M., Ed., Moscow: Transport, 1984. (In Russian.)
14. Zemlyanyy, E.S. and Tektov, M.V., The Implementation of Quorum-Element for Parametric Failure Detection, *Izvestiya Tula State University. Technical Sciences*, 2023, no. 6, pp. 104–109. (In Russian.)
15. Savina, M.G., Musonov, V.M., Khudonogov, V.P., and Seslavin, V.S., The Application of Quorum-Elements for Control of Single Type Sensors, *Reshetnev Readings*, 2013, vol. 1, pp. 377–379. (In Russian.)
16. Dzhandzhgava, G.I., Dyadischev, A.V., and Garifov, R.Sh., On a Concept for Technical Condition Monitoring of Methods Used for Analyzing Physical Media, *Ideas and Innovations*, 2018, vol. 6, no. 3, pp. 64–68. (In Russian.)
17. Kudryashov, B.D., *Osnovy teorii kodirovaniya* (Fundamentals of Coding Theory), St. Petersburg: BKhV-Peterburg, 2016. (In Russian.)
18. Telpukhov, D.V., Demeneva, A.I., Zhukova, T.D., Khrushchev, N.S. The Research and Development of Automation Systems for the Concurrent Error Detection Combinational

Circuits, *Electronic Engineering. Series 3: Microelectronics*, 2018, no. 1 (169), pp. 15–22. (In Russian.)
19. Mytsko, E.A., Osokin, A.N., and Malchukov, A.N., *CRC Checksum. A Study of CRC Checksum Calculation Algorithms*, Saarbrucken: LAP LAMBERT Academic Publishing, 2013.
20. Pavlov, A.A., Tsar'kov, A.N., Sorokin, D.E., et al., Analysis of the Effectiveness of Test-Error Correction of Random Access Memory, *Izvestiya of the Institute of Engineering Physics*, 2015, no. 4 (38), pp. 47–57. (In Russian.)
21. Selivanov, E.P. and Mitin, D.D., Option Method of Construction Noise Proof Hamming Code, *Contemporary Information Technologies*, 2006, no. 3, pp. 34–37. (In Russian.)
22. Pavlov, A.A., Tsarkov, A.N., Romanenko, A.Yu., and Mikheev, A.A., Method of Error Correction in Devices for Transmitting and Processing Information of Telecommunications Systems under the Influence of External and Internal Interference, *Aerospace Instrument-Making*, 2021, no. 8, pp. 13–24. (In Russian.)
23. *GOST* (State Standard) *R 70020-2022: Space Technology. Interfaces and Protocols of High-Speed Device Communication and Complexing of Onboard Spacecraft Systems. SpaceWire-RUS*, Moscow: Russian Institute for Standardization, 2022. (In Russian.)
24. Asokan, R. and Vijayakumar, T., Design of Extended Hamming Code Technique Encryption for Audio Signals by Double Code Error Prediction, *Information Technology and Digital World*, 2021, vol. 3, no. 3, pp. 179–192.
25. Bae, W., Han, J.W., and Yoon, K.J., In-memory Hamming Error-Correcting Code in Mersister Crossbar, *IEEE Trans. on Electron. Devices*, 2022, vol. 69, no. 7, pp. 3700–3707.
26. Zhuravlev, V.G., Kuranova, N.Yu., and Evseeva, Yu.Yu., *Pomekhoustoichivye kody* (Noise-Immune Codes), Vladimir: Vladimir State University, 2013. (In Russian.)
27. Chegis, I.A. and Yablonskii, S.V., Logical Methods of Control of Work of Electric Schemes, Trudy Mat. Inst, Steklov., 1958, vol. 51, pp. 270–360. (In Russian.)
28. *Osnovy tekhnicheskoi diagnostiki* (Fundamentals of Technical Diagnosis), vols. 1 and 2, Parkhomenko, P.P., Ed., Moscow: Energiya, 1976. (In Russian.)

This paper was recommended for publication by V.G. Lebedev, a member of the Editorial Board.

Received November 10, 2024, and revised March 25, 2025. Accepted April 10, 2025.

Author information

Bukov, Valentin Nikolaevich. Dr. Sci. (Eng.), JSC Airborne Aeronautical Systems, Moscow, Moscow, Russia
✉ v_bukov@mail.ru
ORCID iD: https://orcid.org/0000-0002-5194-8251

Bronnikov, Andrei Mikhailovich. Dr. Sci. (Eng.), Bauman Moscow State Technical University, Moscow, Russia
✉ bronnikov_a_m@mail.ru
ORCID iD: https://orcid.org/0009-0009-1216-3521

Popov, Aleksandr Sergeevich. Cand. Sci. (Eng.), Zhukovsky–Gagarin Air Force Academy, Voronezh, Russia
✉ saga30@yandex.ru

Shurman, Vladimir Aleksandrovich. JSC Research Institute of Aviation Equipment, Zhukovsky, Russia
✉ shurman@niiao.ru

Cite this paper
Bukov, V.N., Bronnikov, A.M., Popov, A.S., and Shurman, V.A., Technical Condition Monitoring Methods to Manage the Redundancy of Systems. Part I: Built-in Control and Partition into Classes. *Control Sciences* **2,** 2–10 (2025).

Original Russian Text © Bukov, V.N., Bronnikov, A.M., Popov, A.S., Shurman, V.A., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 3–13.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

SOLVING COMPLEX RESOURCE MANAGEMENT PROBLEMS: FROM CLASSICAL OPTIMIZATION AND GAME THEORY TO MULTI-AGENT TECHNOLOGIES FOR REACHING CONSENSUS

A. V. Leonidov*,** and P. O. Skobelev***,****

*Lebedev Physical Institute, Russian Academy of Sciences, Moscow, Russia
**Moscow Institute of Physics and Technology, Dolgoprudny, Russia
***Samara Federal Research Center, Russian Academy of Sciences, Samara, Russia
****Samara State Technical University, Samara, Russia

*✉ leonidovav@lebedev.ru, **✉ p.skobelev@kg.ru

Abstract. Challenges and complex problems arising in the resource management of modern enterprises are considered. The existing resource planning models, methods, and tools for enterprises are reviewed, and new requirements for adaptive multicriteria resource planning in real time are presented. The concept of autonomous artificial intelligence (AI) systems for adaptive resource planning based on multi-agent technologies is discussed. The evolution of the approach to solving complex resource management problems is described: from traditional optimization of a single objective function, ignoring the individual interests of participants, to game theory with their competition and cooperation. The approach to finding and maintaining a competitive equilibrium (consensus) between participants is further developed via conflict identification and negotiations for conflict resolution with mutual trade-offs. A basic model of a multi-agent demand-supply network with a virtual market and a compensation method for reaching consensus for adaptive resource planning are presented. The functionality and architecture of intelligent adaptive resource planning systems are considered. The implementation results of AI solutions for industrial applications are provided, and the possibility of improving the effectiveness of resource usage by enterprises is shown. Finally, the lessons learned from the experience in R&D work and the prospects of this approach are discussed.

Keywords: resource management, complexity, artificial intelligence, demand-supply networks, autonomous systems, adaptability, multi-agent technologies, self-organization, real-time economics.

INTRODUCTION

The growing complexity of elaborating and realizing optimal decisions in the modern economy is largely explained by the sharp increase in the complexity of demand-supply dynamics, when various perturbing events become a norm rather than an exception [1], and the related need for quick adaptation of enterprises to the changing conditions of economic activity.

At the same time, the growing complexity in enterprise management is, more and more, due to the increasing number and diversity of the objectives and characteristic properties of participants in coordinated decision-making processes with their individual preferences and constraints, e.g., in complex international or national supply chains. Unforeseen events include large-scale ones (the appearance of new major customers, partners, or competitors, the development of new products and technologies, or changes in product supply chains) and day-to-day events (such as equipment failures and delays in operations).

The usual response of company's top managers to poorly predictable business events is to attract additional resources, e.g., hiring new managers and increasing the stock of goods in warehouses and the size of warehouses. Within the traditional decision-making system, the response time to emerging events increases, including the collective elaboration, coordination, adoption, and implementation of decisions. As a consequence, the quality of customer service decreases,

downtimes in the use of resources grow, orders are lost, or costs rise; finally, there is a general reduction in the effectiveness and competitiveness of the business [2].

One reason for this situation is the application of traditional models, methods, and tools for resource planning and optimization with centralized multi-level hierarchical enterprise management and packet data processing. This approach complicates the proper consideration of the individual characteristics, preferences, and constraints of the participants in enterprise management processes, important for the activities carried out; as a result, they are often ignored.

The solution of this problem requires the development of a new paradigm of creating maximally autonomous intelligent resource management systems that make decisions on the current management of enterprise resources instead of a human. This paradigm is oriented towards the emerging real-time network economy with a high level of management autonomy, which, in turn, requires the high adaptability of resource management in case of various unforeseen events [3].

Nowadays, it is becoming possible to solve this problem using artificial intelligence (AI) systems, which operate continuously and can autonomously (independently) make decisions for real-time resource allocation, planning, optimization, monitoring, and control of results, as well as adaptively rearrange the plans based on events.

However, current research projects in the field of AI systems are still mainly focused only on autonomous robots and unmanned aerial and ground vehicles [4]. The ongoing projects in other areas of AI technologies include big and small data analysis, pattern recognition and machine vision, machine learning, etc. Strangely enough, AI technologies for resource management have not yet been included in this list, although using AI for autonomous adaptive management to improve the efficiency of enterprises with increasing order volumes and diversity of attracted resources is a very topical and significant problem.

This paper presents theoretical foundations and practical results of solving complex adaptive resource management problems using AI systems based on the multi-agent technology. Compared to the traditional approaches, this technology allows creating self-organizing schedules of orders and resources with higher openness and flexibility to changes.

In Section 1, the reasons for the growing complexity and dynamics of modern production resource management are investigated. In Section 2, we briefly analyze the limitations of the existing methods and tools for resource planning and optimization, including classical and heuristic optimization methods and methods based on game theory. Section 3 considers the concept of an autonomous AI system for adaptive resource management based on the notion of a multi-agent demand-supply network and a virtual market of program agents for orders, operations, resources, and products. As is shown, the solution of the complex resource management problem can be built by identifying and resolving conflicts through auction-like multi-iteration negotiations using the satisfaction, bonus, and penalty functions of agents and the compensation method in case of mutual trade-offs. Section 4 presents the functionality and architecture of autonomous AI solutions for adaptive resource management. The implementation results of AI solutions for industrial applications are described in Section 5, particularly the possibility of improving the effectiveness of resource usage by enterprises. In Section 6, we discuss the lessons learned from the experience in R&D work on these solutions and their business benefits. The main outcomes of the survey, as well as possible directions of future R&D work in the field of such resource management systems, are outlined in the Conclusions.

1. THE COMPLEXITY OF MODERN RESOURCE MANAGEMENT

Examples of modern resource management problems in enterprises are diverse and may include managing a fleet of trucks, machine shop floors, supply chains, train movements, constellations of satellites and drones, and other applications.

Several examples of such problems have already been considered previously by one of the authors; see [5]. The experience accumulated over the past time allows identifying their main features and formulating, more precisely, the requirements for the approaches applied.

The following key complexity factors are typical of these problems: the large number of daily orders, multi-criteria resource management (maximizing service quality, minimizing financial costs and delivery time, and maximizing profits), an individual approach to orders and resources and their multiple features (shared orders, reusable resources, renewable resources, etc.), the interdependencies between the jobs to be done, the specifics of the resources applied, common or shared costs, flexible or fixed prices, etc.

A main factor in the complexity of resource management is that, in practice, people face many conflicting requirements dictated by many participants in the processes of doing business, from the strategic targets of an entire enterprise to the tactical targets of its de-

partments, as well as the operational targets of executors "on the ground": truck drivers, workers, logisticians, dispatchers, economists, and other employees. According to the common opinion of experienced dispatchers, a good schedule is a well-balanced schedule that considers the preferences and constraints of all participants in each particular situation. Thus, an AI system must generate schedules that, in each situation and at each particular time instant, reflect the balance of many conflicting interests, preferences, and constraints, which is extremely difficult and time-consuming within traditional approaches to resource planning.

Moreover, such schedules are often inhomogeneous, i.e., different fragments of the schedule differ depending on the criteria relevant at a particular time instant, which may change with the arrival of new orders and the occurrence of other events during the computation process. We emphasize that the achieved balance of interests always depends on the development of the situation but refers to a particular time instant. Therefore, at a next time instant, a coordinated "optimal" schedule may lose optimality and even become unrealizable in principle.

This "sliding optimization," actually with harmonizing the interests of all participants in each situation and in real time, requires interactive communication with decision-makers, who can add new events and, moreover, modify their preferences and constraints, approve or reject decisions, and make counter-offers.

In this regard, adaptability should be treated as one of the most important functions of such solutions. It can be defined as the ability of an AI system to rearrange the schedule partially, resolving internal conflicts by negotiations without stopping the system, and maneuver resources flexibly to achieve its goals under uncertainty due to the permanent occurrence of events changing the situation at a priori unpredictable time instants.

2. RESOURCE MANAGEMENT: A REVIEW OF THE EXISTING METHODS AND TOOLS

Traditional packet methods and tools for resource planning and optimization based on linear, dynamic, or constraint programming are well known [6, 7].

However, most of these methods and tools are designed for a problem statement where all orders and resources are known in advance and do not change in real time. Therefore, in the field of enterprise resource planning (ERP), classical package planners offered by SAP, Oracle, Manugistic, i2, ILOG, J-Log, and other companies still dominate the market; in practice, however, these packages tend to implement mainly accounting functions due to the increasing problem dimension, and the built-in modules for resource allocation, planning, optimization, and communication with business participants are of limited application.

To decrease the complexity of combinatorial search, methods with heuristic and metaheuristic rules are practiced to make acceptable decisions in a more reasonable time by reducing the solution search domain [8, 9]:

– greedy local search algorithms based on heuristic rules of a subject matter;

– AI methods based on neural networks and fuzzy logic;

– metaheuristics: genetic algorithms and tabu search;

– simulation, including simulated annealing, etc.;

– stochastic methods such as the Monte Carlo method;

– ant colony and particle swarm optimization algorithms;

– combinations of parallel heuristic optimization algorithms, etc.

However, these methods also use packet processing and do not provide the real-time adaptation of schedules as events occur.

Direct analysis of the above solutions reveals the following problems:

– There are no models, methods, and tools for adaptive resource management.

– Under changes in problem specifications, it is necessary to revise the methods applied and attract experts to reprogram the system.

– Available systems support centralized management based on top-down commands, without considering the opinions and interests, preferences, and constraints of executors.

– Due to the hierarchical rigidity of the systems, it is impossible to respond to events promptly and flexibly, and the schedules are realigned only partially.

– The systems are internally passive and operate in the packet mode only at the user's request.

– The systems are focused on data rather than corporate subject-matter knowledge necessary for automated decision-making.

– Business processes are excessively standardized and hence ignore the individual preferences and constraints of decision-makers.

The high complexity and dynamics of the problems under consideration make traditional centralized hierarchically organized sequential methods and algorithms of combinatorial search or heuristics inefficient when solving the problem of adaptive resource management (in terms of acceptable quality and time required). This factor restrains the implementation of the AI enterprise management systems in practice.

3. NEW MODELS AND METHODS FOR REACHING CONSENSUS IN ADAPTIVE RESOURCE MANAGEMENT

Multi-agent technologies are a key trend in AI; for example, see the monographs [10, 11] and the papers [12, 13].

Recently, multi-agent technologies have been associated with AI agents and *Large Language Models* (LLM), but the basic property of multi-agent technologies is still the ability to create self-organizing systems where each element makes its own decisions. As a result, such systems are more open to change, flexible, and effective in various complex problems.

In this regard, multi-agent technologies are a possible method for solving optimization problems [14]. In the last decade, new models and methods for the distributed solution of resource planning and optimization problems have been developed on the basis of multi-agent technologies. The description of such models and references to the relevant literature can be found in the reviews [15–20].

Note that the transition to multi-agent technology for adaptive locally optimal scheduling reflects a significant change in the problem paradigm compared to the approach with standard packet optimization technologies, where the solution is constructed by a centralized sequential deterministic algorithm. In contrast, a schedule within the multi-agent approach is a distributed and dynamic object in which the scheduling problem is solved in a non-deterministic way with parallel and asynchronous computation processes evolving over a common data structure, mirroring the state of enterprise resources at any given time instant. In this case, each event initiates a transition process from one non-equilibrium state to another, which is realized by the partial adaptive rearrangement of the schedule of orders and resources; in other words, the revision of previously made decisions and the redistribution of previously distributed orders by resources are allowed.

Thus, the problem is to rearrange promptly the schedule in a finite time, which defines the characteristics of the target space of system states achievable within a given period from a given initial state by the method under consideration.

The idea of using models and methods based on agents' self-organization in resource management looks very attractive for software developers. Many useful properties of such algorithms are well studied: they are intuitive, able to cover the individual criteria, preferences, and constraints of all participants, reliably correct, naturally parallelizable, deployable in distributed systems, (in many cases) stable to changes in the problem specification, etc. Of particular interest is the systematic comparison of the results of multi-agent and packet optimization approaches, presented, e.g., in [21, 22]. It is necessary for "marking" the effectiveness of multi-agent algorithms depending on the characteristic modes of the problem.

In general, the architecture of multi-agent distributed optimization models is divided into two large classes: models with autonomous agents and models with additional participation of intermediary agents (mediators). A key element of multi-agent technology is a negotiation protocol that ensures that the process of reaching an agreement between program agents of demand (e.g., necessary actions) and supply (resources) is initiated and evolves. In models with autonomous agents, the latter act independently; in models with mediators, the limited control intervention of agents is possible.

Most of the works use different versions of the *Contract Net Protocol* [23, 24], which regulates the process of submitting and analyzing requests. The discussion of such protocols and their comparative analysis were presented in [25].

The supply-demand balance protocol is implemented using a market pricing mechanism that implies the existence of internal virtual money. Thus, multi-agent models realize the concept of a virtual market (VM), where agents iteratively negotiate, concluding and revising contracts among themselves as well as exchanging jobs and money. Each agent begins the solution search with some initial set of jobs, possibly empty, and then enters into a process of negotiating new solutions. An important part of the solution search is the joint consideration of planning and scheduling. This issue was studied in the survey [16] and the publications cited therein. As a result, an optimal schedule is searched within the process of dynamic self-organization in a network of agents; for models with autonomous agents, the ultimate goal of this process, from a general theoretical point of view, is to reach the state of competitive equilibrium (consensus) in which none of the agents will further improve the result for the entire system. As noted above, the key factor of quality is the finite time to obtain the solution and, as a consequence, the possible difference between the solution obtained in this time and the optimal one.

The concept of a virtual market implemented in multi-agent models fits naturally into the general concept of formulating optimization problems in terms of the virtual economy of interacting agents [26–28], particularly within game theory [29–32]

For models with autonomous agents, game theory underlies the general analysis of possible outcomes of interaction among such agents, including the analysis of game-theoretic Nash equilibria in multi-agent systems (MASs) and algorithms for finding them.

For the scheduling problem, Nash equilibria were analyzed, in particular, in [19, 33, 34]. The main result is a formal proof that finding Nash equilibria in such problems is NP-hard. Note that the problems under consideration are not exceptional in this sense: except for a narrow group of special problems, the search for Nash equilibria in pure strategies belongs at least to the complexity class of PPAD ("*Polynomial Parity Arguments on Directed graphs*") [11]. At the qualitative level, this means that the time to construct a solution is exponential in the parameter(s) reflecting the system heterogeneity. Under a limited time for building a modified schedule, it may be impossible to find the corresponding Nash equilibrium.

Clearly, the NP-hardness of game-theoretic equilibria emphasizes the significance of time constraints in scheduling. An important circumstance, potentially decisive for the classification of the corresponding modes, is the existence of a phase transition by the computational cost of solving NP-hard problems, in particular, scheduling problems [35, 36], that separates phases with easy- and hard-to-find solutions.

As for game-theoretic equilibrium search algorithms, the situation becomes even more complicated: no *universal* algorithms of this kind have been developed to date. For example, the existence of configurations in which no solution can be found was demonstrated [39] for one of the most natural and attractive equilibrium search algorithms based on auction theory [11, 37, 38]. As is also known, in some cases, competitive multi-agent models yield no satisfactory solution, reaching a *deadlock* [40]. The deadlock reflects the insufficiency of a certain protocol used in competitive MASs to resolve conflicts. In this regard, MASs with *mediators* [17, 18] are of significant interest. The general multi-agent architecture with mediators was described in [41]: in addition to the basic competitive layer of agents, it includes mediators that can be addressed by competitive agents to resolve conflicts. In contrast to autonomous agents, mediators have access to a significantly larger amount of information, allowing for more precise planning of dynamic rescheduling. According to the comparison of competitive and mediated architectures [42–44], the introduction of mediators can improve the indicators of solution quality. Various multi-agent models with mediators were considered in [40, 44–48].

An essential issue of MAS architecture design is the analysis of hierarchical and holonic architectures, which was discussed in [49–51].

From the theoretical point of view, in addition to the above interaction architectures, cooperative models are of significant interest, in which the interaction protocol of agents is described within cooperative game theory [11, 41, 44]. In particular, this interest is related to that, except for special cases, competitive equilibrium in game theory is inefficient in terms of overall solution quality. In the current context, this problem was reflected in the review [47].

Since 1999, a similar software development approach for implementing multi-agent solutions of optimization problems was elaborated within the projects described in the monograph [5]. In particular, the attractive properties of such algorithms were already manifested in the first multi-agent prototype of the system for a Volkswagen plant to supply and replace wooden parts for the interior design of luxury cars. The problem was that an expensive car ready for delivery often failed quality control due to deviations of the color or pattern of wooden interior parts from the standard. Such a car was driven to the parking lot, and it took a long time to find, deliver, and mount the new part (significant costs). Note that the SAP production system required 12 to 24 hours to resolve the problem; in practice, the plant's department heads simply called each other and settled the issue through negotiations. It was necessary to develop a system that would quickly and adaptively rearrange the schedule using SAP data. The resulting MAS allowed solving the problem within a few seconds (up to a minute).

In the next period, the multi-agent technology was refined according to the concept of holonic systems: the basic agents of products, resources, and orders and the staff agent (or the agent of the entire system) were implemented within the PROSA reference architecture [48]. Further, the technology took the important step of detailing agents to the level of business-process agents and each individual job; also, the classes and roles of agents were introduced that form multi-agent demand-supply networks representing self-organizing schedules with proactivity and mutual compensations in conflict resolution. For the agents of DS networks, an adaptive decision-making method with compensations under the mutual trade-offs of orders and resources in the virtual market based on satisfaction and bonus-penalty functions was proposed to provide elastic decision-making when resolving conflicts and reaching a new consensus among such agents [52–54].

In the method developed, the agents of orders and resources, as well as those of jobs and products, first select the best conflict-free alternatives and then resolve conflicts until the system is balanced to a new consensus and none of the new alternatives can improve the overall goal function of the system (e.g., profit).

This process reflects the existing practices of experienced managers and dispatchers who generate complex schedules by resolving conflicts and balancing the conflicting interests of all parties to the decision-making process.

The formal problem statement and the description of the method were given in [55].

Recently, the interest in AI systems for enterprise management has increased significantly due to the massive adoption of electronic maps, ERP systems, and the Internet of Things, cell phones, and other devices that translate business into a digital model reflecting the state of resources in real time [56–59]. Here, AI capabilities are mainly associated with prediction, planning, and knowledge extraction during learning, in combination with classical resource planning and optimization and various heuristics. The problem of building a dynamic self-organizing schedule with a prompt, flexible, and efficient rearrangement based on real-time events has not been formulated so far.

4. THE FUNCTIONALITY AND ARCHITECTURE OF THE SOLUTION

The functionality of AI systems for adaptive resource management aims to support the full cycle of autonomous resource management, including:

- collecting new events via sensors, external systems, and mobile devices;
- distributing orders among resources by identifying the most appropriate ones;
- planning orders and resources, i.e., calculating the best possible sequence and determining the start and end time of a job (operation) to fulfill orders;
- optimizing orders and resources (if time is available), i.e., continuously improving the goal functions of all agents involved in resource management;
- predicting new events (new orders or failures) that will be handled as virtual events for the preliminary dynamic reservation of critical resources;
- implementing online communication with users: approving system recommendations, changing preferences or making counter-offers, correcting facts, etc.;
- monitoring and controlling plan fulfillment, i.e., comparing planned and factual results, identifying gaps, and initiating a re-planning event for top management;
- adaptive re-planning in case of a growing gap between the plan and reality (e.g., if the user ignores recommendations and exceeds the time limits);
- experience-based learning, i.e., clustering of events, comparing the planned and factual job completion times (e.g., for analyzing employee productivity);
- real-time "what-if" simulation (multiple simulation lines can be run in parallel with the main plan trajectory to explore the future in real-time);
- evolutionary restructuring of the business network, i.e., generating suggestions to improve the quality and efficiency of operations (selecting a better storage space, etc.).

The approach developed can be generalized to the concept of Smart Solution as an autonomous system for real-time intelligent resource management with the following types of users (Fig. 1) [60, 61]:

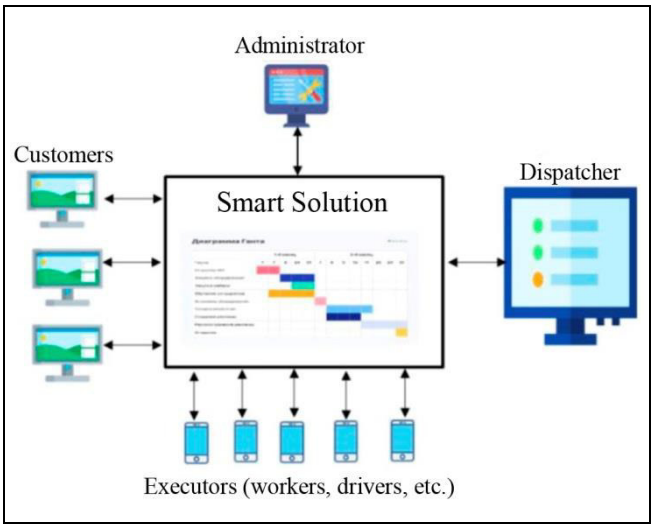

Fig. 1. The concept of an autonomous intelligent resource management system.

– customers, who specify necessary orders, coordinate incoming offers, and further observe the step-by-step fulfillment of their orders;

– dispatchers, who specify planning criteria and approve system-built plans, correct results, and settle the remaining issues;

– executors, who receive shift targets and mark up their fulfillment (when necessary) and introduce unforeseen events causing the adaptive change of plans.

– administrators, who generate logins and passwords for user authorization, manage system databases, etc.

The main types of Smart Solution users and their capabilities may vary depending on the application, but the above basic functionality remains the same for different modifications.

The Smart Solution architecture has the following main components (Fig. 2):

– web systems of users, which are intended to support the business processes of user work;

– an ontology-driven knowledge base, which contains formalized knowledge (classes of concepts and relations) to support real-time decision-making;

– an onto-MAS, which is an ontologically customizable MAS for real-time resource management

– integration, which consists of integration modules with traditional accounting systems (1C, etc.).

The decisions made (in the form of current plans, instructions, or commands) are transmitted to the cell

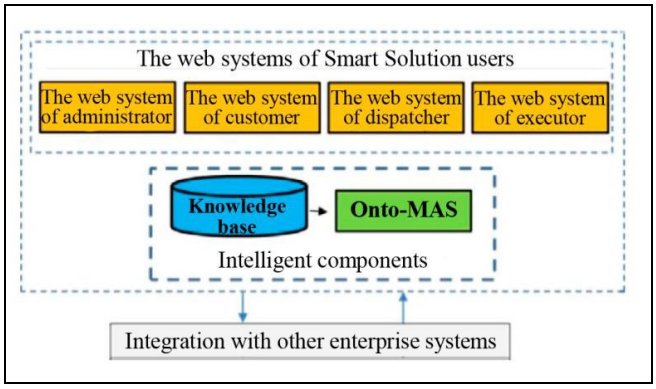

Fig. 2. The main components of the autonomous intelligent system for real-time resource management.

phones of executors or enterprise equipment, with requesting acceptance or confirmation; they can be adaptively revised at any time instant via system-generated or user-entered events if the situation changes.

5. THE RESULTS OF INDUSTRIAL IMPLEMENTATIONS

Based on the above approach, 15 industrial prototypes and full-scale MASs for adaptive resource management were developed between 2000 and 2008, including tanker management, corporate taxi, freight transportation with consolidation, and quite a few different prototypes and small applications (adaptation of a meal plan or workout plan, ordering of household goods, etc.).

During the development and implementation of the systems under consideration, a methodology was elaborated to assess improvements in the effectiveness of resource usage. This methodology evaluates two types of costs:

- direct costs (the reduced time of transporting goods or executing production orders, the decreased use of materials, machinery and machine tools, the reduced wages paid to workers, etc.) and
- overhead costs (the reduced staff of the enterprise (low-level managers, logisticians, dispatchers, economists, etc.).

The economic effect calculated also includes the decreased complexity and labor intensity of management operations, the reduced time of processing unforeseen events, the reduced costs of personnel training, etc.

The problem statement and implementation results were described in detail in [5]. Here, we summarize the main business benefits:

– the increased number of completed orders with the same or reduced resources;
– the reduced order completion time;
– the reduced annual downtime per resource;
– the increased effectiveness of resource usage;
– the formal and systematic knowledge of the subject matter used in decision-making;
– the reduced amount of penalties and fines for delayed order fulfillment;
– the reduced complexity and labor intensity of work for dispatchers, managers, logisticians, and economists;
– the reduced costs of management personnel training.

With these advantages, investment in the systems under consideration is returned on average in three months to one year.

Some of the solutions were used as simulation and decision support tools; however, most have been fully implemented and are still in operation.

At the next stage (2009–2024), the approach was significantly improved and extended from manufacturing and transportation enterprises to new areas of management, particularly the management of passenger and freight railway trains, satellite constellations, beverage supply chains, coal railway car distribution, and other types of resources.

We highlight the additional business benefits identified at this stage (for details, see [62–64]:

– the reduced costs of order execution;
– the reduced number of managers;
– the increased speed and flexibility of decision-making;
– the possibility of business development simulation simultaneously with operational management.

At the first stage of implementation, many additions are made to the knowledge base, which are revealed only when the resource planning results of the system are compared with the work of practitioners. When the quality of decisions made by the system exceeds 50% compared to humans (i.e., the AI system makes more correct decisions than experienced users), we can talk about the beginning of the transition to autonomous AI for “unmanned” enterprise management.

The main result of this period was a more seamless integration of adaptive resource planning and optimization capabilities with monitoring and control of order execution, enabling the creation of “digital twins” of enterprise departments operating in parallel and asynchronously with enterprises and synchronized with them by real-time events.

While final management decisions are still being offered to users for their agreement and approval, the growing trend of gradual transition to autonomous systems designed for the above unmanned management is already visible.

On average, the theoretically proven and confirmed effect from implementing the systems under

consideration may reach 15–40% [62], allowing enterprises to execute more orders with the same amount of production resources (i.e., significantly increasing their efficiency).

6. LESSONS LEARNED AND KEY BENEFITS

The above analysis has revealed several problems arising in the practical implementation of AI enterprise management systems:

• The development of such AI systems needs the participation of highly qualified experts and programmers, takes a lot of time, requires extensive testing, etc.

• The development of self-organizing solutions for business users is a challenging task:

– It is often difficult to assess the "distance" between the result obtained by the system and the "optimal" solution.

– The results depend on the history of the events.

– Small changes lead to an unexpectedly large response (the "butterfly effect").

– The response of the system may slow down in case of transition between equilibrium states.

– If the system is restarted, the planning result may differ.

– Interaction with users becomes more complex and dynamic in the real-time mode.

– The solution is sometimes difficult to explain to the user (the loss of causality), etc.

• Enterprise resource management is business-critical, so this area is still very conservative in adopting new AI solutions.

• Much of the corporate knowledge for decision-making is usually not realized and hidden in the heads of experts; identifying and formalizing this knowledge requires direct communication with dispatchers, engineers, workers, drivers, etc.

• Much of the effort is related to the development of network user interfaces, which must be customizable and inexpensive.

• For a wider range of small- and medium-sized enterprises, further evolution seems to run toward the development of digital SaaS (*Software as a* Service) platforms for an ecosystem of services and additional solutions that can be integrated with existing systems.

In practice, these difficulties are manageable but require special tools for the initial analysis of customer data and integration with (often) out-of-date systems containing possibly irrelevant and incorrect data.

The above difficulties are compensated for by the advantages of Smart Solution, as they:

– improve the effectiveness of resource usage by passing to real-time decision-making;

– solve complex planning problems by replacing combinatorial search with conflict analysis and reaching consensus;

– provide adaptive re-planning with prompt response to events;

– offer a personalized approach to every order, job, product, and resource;

– support active two-way interaction with users for coordinated teamwork;

– reduce the role of the human factor in decision-making;

– reduce development costs by reusing the code in new applications;

– simulate the "if-then" scenario and make predictions to improve decisions;

– create a new digital platform to support business growth without proportionate growth in management staff.

The R&D results can be applied in a wide range of resource management problems within the Industry 5.0 and Society 5.0 concepts, which are oriented to knowledge digitalization and transition to autonomous collective intelligence systems [65].

CONCLUSIONS

A new class of autonomous intelligent systems for unmanned enterprise resource management opens up new opportunities for raising the efficiency of business management, improving customer satisfaction, making businesses more flexible, lowering order execution costs, and reducing lead times and risks.

Time constraints on the elaboration of optimal decisions require the theoretical understanding and revision of existing approaches. In fact, the matter concerns the development of a new methodology of "guided self-organization" and "smart optimization" for elaborating quasi-optimal solutions of exponentially difficult problems with constraints under which the system independently assesses the results and decides on the completion of calculations or the branches of optimization to be further investigated.

The industrial applications developed prove that the multi-agent technology is able to solve a wide range of resource management problems under high uncertainty, complexity, and dynamics. Adaptive resource management helps to increase business efficiency, reduce response time, and improve the quality of service for new orders, as well as raise the effectiveness of resource usage.

As expected, the next step is to create a digital network-centric platform and an ecosystem of digital twins of enterprises to solve complex multi-level resource management problems of large industrial enterprises, transportation and service companies, etc.

As it seems, future work will combine adaptive planning with experience-based learning using neural networks and user interaction based on LLM to build an enterprise knowledge base and to organize a natural language dialog with users, also capable of explaining and harmonizing decisions.

Acknowledgments. *The work of P.O. Skobelev was supported by the Ministry of Science and Higher Education of the Russian Federation.*

REFERENCES

1. *Capitalizing on Complexity? Insights from the Global Chief Executive Officer Study*, USA: IBM, 2010. URL: http://www-935.ibm.com/services/us/ceo/ceostudy2010/index.html. (Accessed January 4, 2025.)
2. Skobelev, P. and Trentesaux, D., Disruptions Are the Norm: Cyber-Physical Multi-Agent Systems for Autonomous Real Time Resource Management, in *Service Orientation in Holonic and Multi-agent Manufacturing*, Studies in Computational Intelligence, Borangiu, T., Trentesaux, D., Thomas, A., et al., Eds., Springer, vol. 6942017, pp. 287–294.
3. *GARTNER. Top Strategic Predictions for 2016 and Beyond: The Future Is a Digital Thing*, Stamford: Gartner, Inc., 2015. URL: https://www.gartner.com/en/documents/3142020. (Accessed January 5, 2025.)
4. Perez, J.A., Deligianni, F., Ravi, D., and Yang, G.-Z., Artificial Intelligence and Robotics, *UK-RAS White Paper*, London: Imperial College, 2017. URL: https://arxiv.org/pdf/1803.10813. (Accessed January 5, 2025.)
5. Rzevski, G. and Skobelev, P., *Managing Complexity*, London–Boston: WIT Press, 2014.
6. *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, Leung, J.Y.-T., Ed., London–New York: Chapman & Hall/CRC, 2004.
7. Vos, S., Meta-heuristics: The State of the Art. Local Search for Planning and Scheduling, in *Lecture Notes in Computer Science*, Nareyek, A., Ed., Berlin: Springer-Verlag, 2001, vol. 2148, pp. 1–23.
8. Binitha, S. and Sathya, S.S., A Survey of Bio inspired Optimization Algorithms, *International Journal of Soft Computing and Engineering*, 2012, vol. 2, no. 2, pp. 137–151.
9. *Handbook of Constraint Programming*, Rossi, F., Van Beek, P., and Walsh, T., Eds., Amsterdam: Elsevier, 2006.
10. Wooldridge, M., *An Introduction to Multi-Agent Systems*, Hoboken: John Wiley & Sons, 2009.
11. Shoham, Y. and Leyton-Brown, K., *Multi-agent Systems: Algorithmic, Game Theoretic and Logical Foundations*, Cambridge: Cambridge Univ. Press, 2009.
12. Slovokhotov, Yu.L., and Novikov, D.A., Distributed Intelligence of Multi-Agent Systems. Part I: Basic Features and Simple Forms, *Control Sciences*, 2023, no. 5, pp. 2–17.
13. Slovokhotov, Yu.L. and Novikov, D.A., Distributed Intelligence of Multi-Agent Systems. Part II: Collective Intelligence of Social Systems, *Control Sciences*, 2023, no. 6, pp. 2–17.
14. Davidsson, P., Persson, J., and Holmgren, J., On the Integration of Agent-Based and Mathematical Optimization Techniques, in *Agent and Multi-Agent Systems: Technologies and Applications*, KES-AMSTA 2007, Lecture Notes in Computer Science, Nguyen, N.T., Grzech, A., Howlett, R.G., and Jain, L.C., Eds., Berlin–Heidelberg: Springer, 2007, vol. 4496, pp. 1–10.
15. Shen, W. and Norrie, D.H., Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey, *Knowledge and Information Systems*, 1999, vol. 1, pp. 129–156.
16. Shen, W., Wang, L., and Qi, H., Agent-Based Distributed Manufacturing Process Planning and Scheduling: a State-of-the-Art Survey, *IEEE Transactions on Systems, Man and Cybernetics*, 2006, vol. 36, pp. 563–577.
17. Quelhadj, D. and Petrovich, S., A Survey of Dynamic Scheduling in Manufacturing Systems, *Journal of Scheduling*, 2009, vol. 12, pp. 417–431.
18. Barbati, M., Bruno, M., and Genovese, A., Applications of Agent-Based Models for Optimization Problems: A Literature Review, *Expert Systems with Applications*, 2012, vol. 39, pp. 6020–6028.
19. Agnestis, A., Multiagent Scheduling Problems, *INFORMS Tutorials in Operational Research*, 2014, pp. 151–170.
20. Lin, G.Y.-J. and Solberg, J.J., Integrated Shop Floor Control Using Autonomous Agents, *IIE Transactions*, 1992, vol. 24, pp. 57–71.
21. Frey, D., Nimis, J., Worn, H., and Lockemann, P., Benchmarking and Robust Multi-agent-Based Production and Control, *Engineering Applications of Artificial Intelligence*, 2003, vol. 16, pp. 307–320.
22. Mes, M., van der Heijden, M., and van Harten, A., Comparison of Agent-Based Scheduling to Look-Ahead Heuristics for Real-Time Transportation Problems, *European Journal of Operational Research*, 2007, vol. 181, pp. 59–75.
23. Smith, R.G., The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Transactions on Computers*, 1980, vol. 29, pp. 1104–1113.
24. Davis, R. and Smith, R.G., Negotiation as a Metaphor for Distributed Problem Solving, *Artificial Intelligence*, 1983, vol. 20, pp. 63–109.
25. Reaidy, J., Masotte, P., and Diep, D., Comparison of Negotiation Protocols in Dynamic Agent-Based Manufacturing Systems, *International Journal of Production Economics*, 2006, vol. 99, pp. 117–130.
26. Ferguson, D., Yemini, Y., and Nikolaou, C., Microeconomic Algorithms for Load Balancing in Distributed Computer Systems, *Proceedings of the 8th International Conference on Distributed Computing Systems*, San Jose, 1988, pp. 491–499.
27. Waldspurger, C.A., Hogg, T., Huberman, B.A., et al., Spawn: A Distributed Computational Economy, *IEEE Transactions on Software Engineering*, 1992, vol. 18, pp. 103–117.
28. Huberman, B.A. and Hogg, T., Distributed Computation as an Economic System, *Journal of Economic Perspectives*, 1995, vol. 9, pp. 141–152.

29. Wang, J., Hong, Y., Xu J., et al., Cooperative and Competitive Multi-Agent Systems: From Optimization to Games, *IEEE/CAA Journal of Automatica Sinica*, 2022, vol. 9, pp. 763–783.
30. Renna, P., A Review of Game Theory Models to Support Production Planning, Scheduling, Cloud Manufacturing and Sustainable Production Systems, *Designs*, 2024, vol. 8, no. 2, art. no. 26.
31. Yang, B. and Johansson, M., Distributed Optimization and Games: A Tutorial Overview, in *Networked Control Systems*, Lecture Notes in Control and Information Sciences, Bemporad, A., Heemels, M., and Johansson, M., Eds., London: Springer, 2010, vol. 406, pp. 109–148.
32. Madsen, J.R. and Shamma, J.S., Game Theory and Distributed Control, in *Handbook of Game Theory and Applications*, 2015, vol. 4, pp. 861–899.
33. Briand, C., Ngueveu, S.U., and Sucha, P., Finding an Optimal Nash Equilibrium to the Multi-agent Scheduling Problem, *Journal of Scheduling*, 2017, vol. 20, pp. 475–491.
34. Agnetis, A., Briand, C., Ngueveu, S.U., and Sucha, P., Price of Anarchy and Price of Stability in Multi-agent Project Scheduling, *Annals of Operations Research*, 2020, vol. 285, pp. 97–119.
35. Hogg, T., Huberman, B.A., and Williams, C.P., Phase Transitions and the Search Problem, *Artificial Intelligence*, 1996, vol. 81, pp. 1–15.
36. Herroelen, W. and De Reyck, B., Phase Transitions in Project Scheduling, *Journal of the Operational Research Society*, 1999, vol. 50, pp. 148–156.
37. Easley, D. and Kleinberg, J., Networks, Crowds, and Markets: Reasoning about a Highly Connected World, Cambridge: Cambridge Univ. Press, 2010. URL: http://www.cs.cornell.edu/home/kleinber/networks-book/. (Accessed January 6, 2025.)
38. Wellman, M., Walsh, W.E., Wurman, P., and Makkie-Mason, K., Auction Protocols for Decentralized Scheduling, *Games and Economic Behavior*, 2001, vol. 35, pp. 291–303.
39. Hall, N.G. and Liu, Z., On Auction Protocols for Decentralized Scheduling, *Games and Economic Behavior*, 2011, vol. 72, pp. 583–585.
40. Chen, W., Maturana, F., and Norrie, D.H., MetaMorph II: An Agent-Based Architecture for Distributed Intelligent Design and Manufacturing, *Journal of Intelligent Manufacturing*, 2000, vol. 11, pp. 237–251.
41. Munich, L., Schedule Situations and Their Cooperative Game Theoretic Representations, *European Journal of Operational Research*, 2024, vol. 316, pp. 767–778.
42. Cavalieri, S., Garetti, M., Macchi, M., and Taisch, M., An Experimental Benchmarking of Two Multi-agent Architectures for Production Scheduling and Control, *Computers in Industry*, 2000, vol. 43, pp. 139–152.
43. Brennan, R.W. and Norrie, D.H., Evaluating the Performance of Reactive Control Architectures for Manufacturing Production Control, *Computers in Industry*, 2001, vol. 46, pp. 235–245.
44. Messie, D. and Oh, J.C., Cooperative Game Theory within Multi-agent Systems for Systems Scheduling, *Proceedings of the 4th International Conference on Hybrid Intelligent Systems (HIS'04)*, Kitakyushu, Japan, 2004, pp. 166–171.
45. Ramos, C., An Architecture and a Negotiation Protocol for the Dynamic Scheduling of Manufacturing Systems, *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, 1994, vol. 4, pp. 3161–3166.
46. Maturana, F., Shen, W., and Norrie, D.H., MetaMorph: An Adaptive Agent-Based Architecture for Intelligent Manufacturing, *International Journal of Production Research*, 1999, vol. 37, pp. 2159–2173.
47. Paccagnan, D., Chandan, R., and Marsden, J.R., Utility and Mechanism Design in Multi-agent Systems: An Overview, *Annual Reviews in Control*, 2022, vol. 53, pp. 315–328.
48. Brussel, H.V., Wyns, J., and Valckenaers, P., Reference Architecture for Holonic Manufacturing Systems: PROSA, *Computer in Industry*, 1998, vol. 37, no. 3, pp. 255–274.
49. Bongaerts, L., Monostori, L., Mcfarlane, D., and Kadar, B., Hierarchy in Distributed Shop Control, *Computers in Industry*, 2000, vol. 43, pp. 123–137.
50. Rabelo, R.J. and Camarinha-Matos, L.M., Negotiation in Multi-agent Based Dynamic Scheduling, *Robotics & Computer-Integrated Manufacturing*, 1994, vol. 11, pp. 303–309.
51. Gou, L., Luh, P.B., and Kyoya, Y., Holonic Manufacturing Scheduling: Architecture, Cooperation Mechanism, and Implementation, *Computers in Industry*, 1998, vol. 37, pp. 213–231.
52. Skobelev, P., Open Multi-agent Systems for Decision-Making Support, *Avtometriya*, 2002, no. 6, pp. 45–61. (In Russian.)
53. Skobelev, P. and Vittikh, V., Multiagent Interaction Models for Constructing the Needs-and-Means Networks in Open Systems, *Automation and Remote Control*, 2003, vol. 64, no. 1, pp. 162–169.
54. Vittikh, V. and Skobelev, P., The Compensation Method of Agents Interactions for Real Time Resource Allocation, *Avtometriya*, 2009, no 2, pp. 78–87. (In Russian.)
55. Skobelev, P., Multi-Agent Systems for Real Time Adaptive Resource Management, in *Industrial Agents: Emerging Applications of Software Agents in Industry*, Leitão, P. and Karnouskos, S., Eds., Amsterdam: Elsevier, 2015, pp. 207–230.
56. Peretz-Andersson, E., Tabares, S., Mikalef, P., and Parida, V., Artificial Intelligence Implementation in Manufacturing SMEs: A Resource Orchestration Approach, *International Journal of Information Management*, 2024, vol. 77, no. 1, art. no. 102781. DOI: 10.1016/j.ijinfomgt.2024.102781
57. Dwivedi, Y.K., et al. Artificial Intelligence (AI): Multidisciplinary Perspectives on Emerging Challenges, Opportunities, and Agenda for Research, Practice and Policy, *International Journal of Information Management*, 2021, vol. 57, art. no. 101994.
58. Yang, W., Li, W., Cao, Y., et al., An Information Theory Inspired Real-Time Self-Adaptive Scheduling for Production-Logistics Resources: Framework, Principle, and Implementation, *Sensors*, 2020, vol. 20, art. no. 7007. DOI: 10.3390/s20247007
59. Mourtzis, D., Advances in Adaptive Scheduling in Industry 4.0, *Front. Manuf. Technol.*, 2022, vol. 2, art. no. 937889. DOI: 10.3389/fmtec.2022.937889
60. Leitão, P., Colombo, A., and Karnouskos, S., Industrial Automation Based on Cyber-physical Systems Technologies:

Prototype Implementations and Challenges, *Computers in Industry*, 2016, vol. 81, pp. 11–25.

61. Gorodetsky, V.I., Laryukhin, V.B., and Skobelev, P.O., Conceptual Model of a Digital Platform for Cyber-Physical Management of Modern Enterprises Part 1. Digital Platform and Digital Ecosystem, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2019, vol. 20, no. 6, pp. 323–332. https://doi.org/10.17587/mau.20.323-332. (In Russian.)
62. Rzevski, G., Skobelev, P., and Zhilyaev, A., Emergent Intelligence in Smart Ecosystems: Conflicts Resolution by Reaching Consensus in Resource Management, *Mathematics*, 2022, vol. 10, no. 11, art. no. 1923.
63. Galuzin, V., Galitskaya, A., Grachev, S., et al., The Autonomous Digital Twin of Enterprise: Method and Toolset for Knowledge-Based Multi-Agent Adaptive Management of Tasks and Resources in Real Time, *Mathematics*, 2022, vol. 10, no. 10, art. no. 1662.
64. Grachev, S.P., Zhilyaev, A.A., Laryukhin, V.B., et al., Methods and Tools for Developing Intelligent Systems for Solving Complex Real-Time Adaptive Resource Management Problems, *Automation and Remote Control*, 2021, vol. 82, pp. 1857–1885.
65. Skobelev, P.O. and Borovik, S.Y., On the Way from Industry 4.0 to Industry 5.0: From Digital Manufacturing to Digital Society, *Industry 4.0*, 2017, vol. 2, no. 6, pp. 307–311.

This paper was recommended for publication by A.A. Lazarev, a member of the Editorial Board.

Received January 27, 2025, and revised April 24, 2025. Accepted April 29, 2025.

Author information

Leonidov, Andrei Vladimirovich. Dr. Sci. (Phys.–Math.), Lebedev Physical Institute, Russian Academy of Sciences, Moscow, Russia; Moscow Institute of Physics and Technology, Dolgoprudny, Russia
✉ leonidovav@lebedev.ru
ORCID iD: https://orcid.org/0000-0002-6714-6261

Skobelev, Petr Olegovich. Dr. Sci. (Eng.), Samara Federal Research Center, Russian Academy of Sciences, Samara, Russia; Samara State Technical University, Samara, Russia
✉ p.skobelev@kg.ru
ORCID iD: https://orcid.org/0000-0003-2199-9557

Cite this paper
Leonidov, A.V. and Skobelev, P.O., Solving Complex Resource Management Problems: From Classical Optimization and Game Theory to Multi-Agent Technologies for Reaching Consensus. *Control Sciences* **2,** 11–21 (2025).

Original Russian Text © Leonidov, A.V. and Skobelev, P.O., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 14–26.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

HOW DOES THE INTERNAL STRUCTURE OF A COMPLEX SYSTEM INFLUENCE ITS OVERALL RISK? RISK MINIMIZATION FOR TREES

A. A. Shiroky* and A. O. Kalashnikov**

*,**Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

*✉ shiroky@ipu.ru, **✉ aokalash@ipu.ru

Abstract. The Defender–Attacker problem is often employed as a mathematical framework in risk management. In this problem, the above players with opposite goals allocate limited resources to system elements to minimize or maximize a risk function. It has been well-studied under the assumption of independent system elements. However, in complex systems, elements interact, causing significant differences between the measured and predicted risks. Although models with the interdependence of system elements are regularly considered in the literature, no comprehensive understanding has been formed of how the structure of a complex system influences its overall risk. We address this issue in a series of papers by investigating system structures of increasing complexity. Chains and stars have been analyzed previously; in this paper, the findings are extended to arbitrary trees. We optimize the placement of elements within a tree to minimize risk; derive upper bounds for the relative error of an approximate algorithmic solution of this problem for trees with a few branches and leaves; and explore the dynamics of these bounds when increasing the number of leaves and branches. As demonstrated, the resulting upper bounds do not exceed their counterparts for stars from the previous works.

Keywords: complex systems, risk, system structure, risk management, risk minimization algorithms, the problem of optimal element placement.

INTRODUCTION

The complexity of risk management is primarily connected with its multidisciplinarity. For example, the authors of the book [1] identified 15 *dimensions of* risk management, which include both relatively narrow fields (risk management in supply chains, financial risk management) and global ones (e.g., ethics in risk management). The second part of the book considered six cross-disciplines that partially overlap all fields, namely, *risk culture*, *risk-based decision making*, *risk leadership in complexity*, *resilience*, *communication uncertainty*, and *organizational change management and risk*. This classification is neither complete nor the only possible one. It illustrates that risk management can be discussed in relation to the specifics of a particular controlled system and with application to processes and properties characteristic of whole classes of systems. In this case, the models and methods, terminology, and even the definition of risk used will differ.

Without an accepted universal risk management model, the unifying role is played by basic principles valid for any controlled system. It is reflected in the ISO standard [2], offering a fairly general definition of risk due to uncertainty influencing goal achievement. As noted, the consequence of this influence should be understood as a deviation from the expected result or event (positive and (or) negative). To use such a definition in practice, one has to *measure* goals, uncertainty, and the deviations caused by it. Hence, it is necessary to investigate quantitative relationships for risk management using an appropriate mathematical apparatus.

Based on the character of risk management intended to minimize deviations, the mathematical problem of risk management should belong to the class of optimization problems. (In the case of players with strategic behavior in the system, the problem can be game-theoretic; for example, see [3–5].) However, an attempt to find studies devoted to mathematical models of risk management not related to a particular ob-

ject, class, or field is likely to fail. The reason is that if a research work deals with mathematical models of optimization applied to risk management, it will belong to the corresponding branch of mathematics, not to risk management. If the matter concerns risk management, the controlled object will be described explicitly, which binds the research work to its specifics.

Nevertheless, one may suppose the existence of models universal enough to consider in quantitative terms the general principles of risk management without binding to particular controlled objects, systems, or their classes. As it seems, this criterion is best met by the model of a protected system in the form of a weighted directed graph: the vertices are system elements (arbitrary objects), and the arcs with assigned weights characterize the direction and strength of connections between these elements that are important for risk management.

Note that such a graph is a complex network of arbitrary topology. The control object modeled (a protected system) can be a social network [6], a network of organizations [7], a computer network [8], or even belong to another class. In this paper, we consider a purely mathematical problem statement. In other words, the structure of a protected system does not necessarily reflect exactly the physical or organizational structure of the object modeled. At the same time, the arcs of a corresponding digraph show the mutual influence of elements. For example, when reducing the risks of aviation accidents in a region, the model will reflect rather the structure of causal relations between the types of accidents, their preconditions, and influence factors than the connections between the elements of the air traffic control infrastructure. If all system elements are independent from the viewpoint of risk transfer to each other, an adequate model will be a special case of an edgeless graph.

The general problem of risk management in complex networks can be formulated as follows.

Consider a protected system consisting of a finite set of elements (arbitrary objects): $S = \{s_1,..., s_i,..., s_n\}$, $i \in N = \{1,..., n\}$, $n \in \mathbb{N}$. Suppose the existence of two actors (also arbitrary for the time being), which will be called players *A* and *D* (the *Attacker* and *Defender*, respectively). They have opposite interests regarding the state of system *S*.

Let player *D* possess some resource amount $X \geq 0$ to be allocated, in an arbitrary way, among the elements of system *S*: $x = (x_1,..., x_n)$, $x_i \geq 0$, $i \in N$, $\sum_{i=1}^{n} x_i \leq X$. Similarly, player *A* also has some resource amount $Y \geq 0$ to be arbitrarily allocated among the elements of system *S*: $y = (y_1,..., y_n)$, $y_i \geq 0$, $i \in N$, $\sum_{i=1}^{n} y_i \leq Y$.

The model under consideration involves any measurable and arbitrarily divisible resource that can be represented by a nonnegative real number. Depending on the context, the resource can be capital, labor, time, production capacity, etc. (e.g., costs).

Suppose that the risk transfer influence is described by a weighted digraph $G(S, W)$, $W \subseteq S \times S$, $w_{ij} = (s_i, s_j) \in W$, $i, j \in N$. Let functions

$$\rho : s \to \mathbb{R}^0_+, \sigma : W \to \mathbb{R}^0_+,$$

be defined on $G(S,W)$, where ρ_i, $i \in N$, is the weight of vertices (the current value of local risk) and σ_{ij}, $i, j \in N$, is the weight of arcs (the intensity of risk transfer between system elements). The matrix $\Sigma = \|\sigma_{ij}\|$ represents the degree (or strength) of influence of the *i*th element of *S* on the *j*th one. The initial value of the functions ρ_i at $t = 0$, $\rho_i = \rho_i(x, y, t) = \rho_i(x, y, t = 0)$, is determined by the resource allocations x and y. The subsequent values of the weights ρ_i (for $t > 0$) depend only on the time-preceding values of these functions. Due to the mutual influence of system elements, their weights vary as follows:

$$\rho_i(t+1) = \rho_i(t) + \sum_{k=1}^{n} \sigma_{ik} (\rho_i(t) - \rho_i(t-1)),$$
$$t = 0, 1,...; \ \rho_i(t = 0) = \tilde{\rho}_i. \quad (1)$$

The arguments x and y in the above formula are omitted for the sake of compactness.

Let $\mathcal{X}(X)$ and $\mathcal{Y}(Y)$ denote the sets of admissible allocations of the resource amounts *X* and *Y*, respectively, among the elements of system *S* by players *D* and *A*:

$$\mathcal{X}(X) = \left\{ (x_1,..., x_n) \in \mathbb{R}^n_+ : x_i \geq 0, \ i \in N, \ \sum_{i=1}^{n} x_i \leq X \right\},$$

$$\mathcal{Y}(Y) = \left\{ (y_1,..., y_n) \in \mathbb{R}^n_+ : y_i \geq 0, \ i \in N, \ \sum_{i=1}^{n} y_i \leq Y \right\}.$$

Then, the problem of player *D* (the Defender's problem) is to find a resource allocation $x^* \in X$ minimizing the overall risk (i.e., the risk characterizing the

vulnerability of the entire system). It can be formally written as

$$x^* = \operatorname*{Argmin}_{x\in X} \lim_{t\to\infty} \rho(x,\ y,\ t) = \operatorname*{argmin}_{x\in X} \sum_{i=1}^{n} \lim_{t\to\infty} \rho_i(x,\ y,\ t). \quad (2)$$

This problem with constraints imposed on the eigenvalues of the mutual influence matrix of elements was solved in the paper [8]. For the problem statement under consideration, it is required to identify, with sufficient accuracy, the current values of local risks and the functional dependencies $\rho(x, y, \cdot)$ and, moreover, to quantitatively characterize the mutual influence of local risks. These tasks can be extremely labor-intensive and even impossible for real systems. Therefore, a topical problem is to find general risk management principles for a complex network system in order to achieve risk reduction even under incomplete information. The paper is devoted to solving this problem for trees.

The remainder of this paper is organized as follows. Section 1 briefly reviews mathematical models of failure propagation in complex networks. Next, Section 2 provides a general statement of the risk management problem in a complex system with a tree structure. A suboptimal solution of this problem is proposed in Section 3. The prospects of further research are discussed in the Conclusions.

1. MATHEMATICAL MODELS OF FAILURE PROPAGATION IN COMPLEX NETWORKS: A BRIEF REVIEW

In the most general case, the structure of a complex system can be considered a complex network of arbitrary topology. A successful attack on some element of a system (in other words, its failure) will be comprehended as the occurrence of an event under which this element ceases functioning. For the sake of simplicity, we analyze only the binary case in this paper: an element can be completely functional or nonfunctional. Many models have been developed to investigate various destructive effects (including targeted attacks on vertices and edges) in such networks, and new ones are proposed regularly. Risk assessment models of failure propagation are widely used when studying various complex systems, such as cyber-physical [9–17], computational [18–19], and social-medical [20–22].

Early models described the development of failures caused by non-targeted (e.g., random) influences. Among them, the best-known ones are the error resilience model [23–25], the forest fire model [26–28] and its derivatives, cellular automata-based models [29–32], and percolation models with random attacks [33]. The latter have several modifications in which destructive influences on network vertices and edges are targeted. These include percolations with targeted attacks [34–36], percolations with localized attacks [37–40], and *k*-core percolations [41–43].

The above failure propagation models combine well with the classical models of risk management in complex Defender–Attacker networks [44–46]. Recall that such models describe a conflict between two players (the Defender and Attacker) with opposite goals concerning the system under consideration. The Attacker spends available resources from some limited pool to disable the system. In turn, the Defender attempts to counteract the Attacker. In classical formulations, the Defender optimally allocates the resources among system elements to minimize its overall risk. However, this player can alternatively modify the system structure to achieve the same goal. Other models are required to describe such a scenario.

For example, the models of cascading error propagation [47, 48] cover structural changes, but such changes are not supposed targeted. The possibility of an intentional structure change is envisioned in models modified to the case of two interconnected networks [49–51]; meanwhile, this possibility applies only to the edges connecting the networks to each other.

Thus, the existing modeling apparatus is insufficient to manage the structure of a complex system, including minimization of its overall risk. In this paper and several previous studies, we focus on calculating the influence of the structure on risk, without regard to the resources allocated.

For this purpose, the basic problem (2) has been reformulated: the search for an optimal resource allocation to the elements of a fixed-structure system has been replaced by the search for an optimal placement of elements in some given structure and comparison of the structures. Dealing with this problem head-on seems impractical due to its high computational complexity, so we find approximate solutions for various structures in ascending order of their complexity, namely:

1) simple chain (see the analytical solution in [52]),

2) star (see an approximate solution with a guaranteed error in [53]),

3) tree (considered here),

4) an arbitrary structure (the solution will be constructed by generalizing the results established for simpler structures).

Note that the general efficient management problem of a complex system under uncertainty is equiva-

lent to the problem of risk minimization, where risk is understood as a measurable deviation from the maximum effective (target) mode of functioning of this system [2]. The mathematical equivalence of the problems was shown, e.g., in the paper [54]. No doubt, the problem of synthesizing or improving the structure of a controlled system (in particular, an organizational system) does not come to resource allocation, and risk is only one of the key performance indicators of its functioning. Nevertheless, when achieving the goals of a control system, risk should be considered the most significant indicator.

2. RISK CONTROL IN A COMPLEX SYSTEM WITH A TREE STRUCTURE: PROBLEM STATEMENT

Suppose that a protected system includes n elements $s_1,..., s_n \in S, n \in \mathbb{N}$. Let two numbers be assigned to each element: $p_i^0 \in (0, 1]$, denoting the eigen probability of a successful attack on the ith element, and $u_i > 0, u \in \mathbb{R}^+$, denoting the damage amount inflicted in case of a successful attack on the ith element.

Definition 1. The eigen risk of the ith element is the value $\rho_{s_i}^0 = u_i p_i^0$. ♦

We define a structure $W_m = \langle G(V, E), T\rangle, T \subseteq V$, where $G(V, E)$ is a directed graph with a vertex set V and an arc set E, and T is a subset of V (further called the *perimeter*). This paper considers structures with a perimeter consisting of exactly one vertex.

Definition 2. A vector sequence

$$B = \left\{\left(b_{01},..., b_{0q_0}\right), \left(b_{11},..., b_{1q_1}\right),..., \left(b_{l1},..., b_{lq_l}\right),..., \left(b_{L1},..., b_{Lq_L}\right)\right\},$$
$$b, L, q_l \in \mathbb{N}, l \in \mathbb{N} \cup \{0\},$$

is said to define a directed tree with m leaves if:

- The number $b_{li}, i \in \{1,..., q_l\}$, indicates the number of outgoing arcs for the corresponding vertex.
- The number L is the length of the maximum path.
- The number $q_l, l \le L$, determines the number of vertices at layer l (the path from the tree root to such vertices has length l);
- $q_0 = 1; q_l = \sum_{i=1}^{q_{l-1}} b_{(l-1)i} \ \forall l > 0; q_L = m$;
- $b_{L1} = b_{L2} = ... = b_{Lq_L} = 0$. ♦

Definition 3. The system structure generated by a sequence B is a tree with m leaves, denoted by $W_m = \langle G(V, E), T\rangle$, if

$$V = \left\{\{v_0\} \cup \bigcup_{j=1}^{b_{01}} \{v_{0j}\} \cup \bigcup_{i=1}^{q_1}\bigcup_{j=1}^{b_{1i}} \{v_{0ij}\} \cup ... \cup ... \cup \bigcup_{i=1}^{q_{L-1}}\bigcup_{j=1}^{b_{(L-1)i}} \{v_{0...ij}\}\right\};$$

$$E = \left\{\bigcup_{j=1}^{b_{01}} \{(v_0, v_{0j})\} \cup \bigcup_{i=1}^{q_1}\bigcup_{j=1}^{b_{1i}} \{(v_{0i}, v_{0ij})\} \cup ... \cup ... \cup \bigcup_{i=1}^{q_{L-1}}\bigcup_{j=1}^{b_{(L-1)i}} \{(v_{0...i}, v_{0...ij})\}\right\}; \ T = \{v_0\}. \blacklozenge$$

Figure 1 shows a tree with four leaves at the third layer as one example. Note that a special case of a tree with $b_{li} = 1, q_l = m \ \ \forall l < L, l \ne 0$, is a star with m rays. The corresponding types of structures have been considered previously in [53].

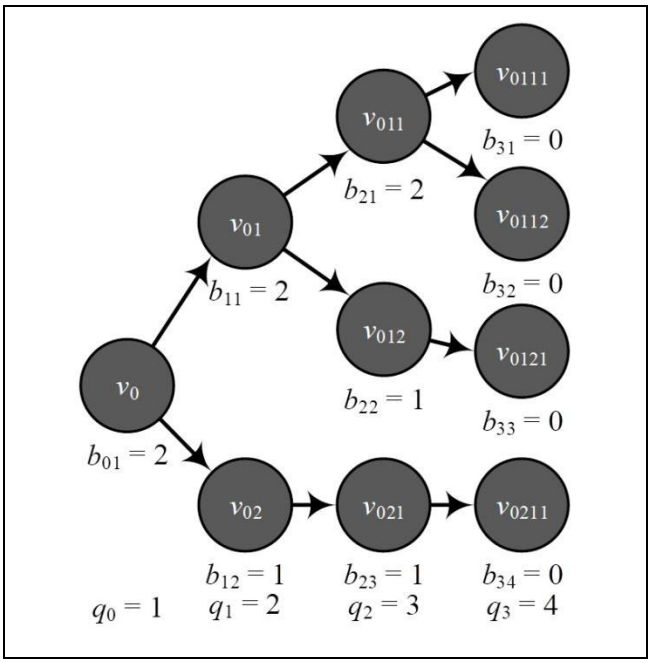

Fig. 1. A tree with m = 4 and L = 3. Vertex numbers are unique and reflect a simple path to the perimeter: it includes all vertices with numbers representing the subrows of the vertex number under consideration.

Definition 4. A bijective mapping $M^{-1}: S \to V$ is called a placement of elements of S in a tree W_m. The corresponding inverse mapping $M: V \to S$ is called the projection of the tree W_m into the set of elements S. ♦

Note that such a mapping exists only if the number of vertices in the graph $G(V, E)$ is equal to the number of elements in the protected system. In the case of an infinite number of vertices, the sets V and S must be countable.

Definition 5. The overall risk of a system with a set of elements S, placed in a tree W_m via a bijective mapping $M^{-1}: S \to V$, is the value

$$\rho\left(S, W_m, M^{-1}\right) = \rho_{M(v_o)} + \sum_{j=1}^{b_{01}} \rho_{M\left(v_{0j}\right)} + \\ + \sum_{i=1}^{q_1} \sum_{j=1}^{b_{1i}} \rho_{M\left(v_{0ij}\right)} + \ldots + \sum_{i=1}^{q_{L-1}} \sum_{j=1}^{b_{(L-1)i}} \rho_{M\left(v_{0\ldots ij}\right)}. \blacklozenge \quad (3)$$

Let the protected system include a set of elements $S = \{s_1, s_2, \ldots, s_n\}$, $n \in \mathbb{N}$, with the corresponding eigen probabilities of successful attack, $P = \left\{p_{s_1}^0, p_{s_2}^0, p_{s_n}^0\right\}$, and damage amounts $U = \left\{u_{s_1}, u_{s_2}, \ldots, u_{s_n}\right\}$. Suppose also that possible attack paths are given by a tree $W_m = \left\langle G(V, E), T\right\rangle$, where $\sum_{l=1}^{L} q_l = n$. Then the overall risk minimization problem of the protected system is to find a placement M^{-1} of elements of S in the structure W_m such that

$$\rho(S, W_m, M^{-1}) \to \min. \quad (4)$$

For the special case $m = 1$, the exact solution has been described in [52]. For the case $q_1 = m \ \forall l < L, l \neq 0$, a suboptimal solution with an a priori bound of the relative error has been obtained in [53]. In this paper, we similarly derive such a bound for trees.

3. THE OPTIMAL PLACEMENT OF ELEMENTS IN A TREE STRUCTURE: AN APPROXIMATE SOLUTION

Suppose that for all system elements, the damage amounts in case of a successful attack are equally estimated: $u_{s_i} = u \ \forall i \in \{1, \ldots, n\}$. Then problem (4) takes the form

$$\rho\left(S, W_m, M^{-1}\right) = u\left(p_{M(v_0)} + \sum_{k=0}^{L-1} \sum_{i=1}^{q_k} \sum_{j=1}^{b_{ki}} \left(p_{M\left(v_{0\ldots ij}\right)} \cdot \right.\right. \\ \left.\left. \cdot p_{M\left(v_{0\ldots i}\right)} \cdot \ldots \cdot p_{M(v_0)} \right)\right) \to \min. \quad (5)$$

In addition, we require that the expression

$$p_{M(v_0)} + \sum_{k=0}^{L-1} \sum_{i=1}^{q_k} \sum_{j=1}^{b_{ki}} \left(p_{M\left(v_{0\ldots ij}\right)} p_{M\left(v_{0\ldots i}\right)} \cdot \ldots \cdot p_{M(v_0)} \right)$$

is finite for any values of L and $m = q_L$. For this purpose, let the eigen risks of all system elements be bounded above by a value called the marginal eigen risk; see the definition below and [53].

Definition 6. The marginal eigen risk of an element of a protected system placed in a structure $W_m = \langle G(V, E), T\rangle$ is the value

$$\rho_{\max}^0 = \frac{u}{1 + \sqrt{m}}. \blacklozenge$$

Note that under the constraint $p_i \le \frac{1}{1+\sqrt{m}} = p_{\max}^0$, we have the inequality

$$\rho\left(S, W_m, M^{-1}\right) \le \\ \le u\left(p_{\max}^0 + \sum_{k=0}^{L-1} \sum_{i=1}^{q_k} \sum_{j=1}^{b_{ki}} \left(p_{\max}^0\right)^{k+1} \right) \le u. \quad (6)$$

In formula (6), equality is achieved at $L = \infty$ for any finite m. Due to the construction of (6), the upper bounds on the increment of the overall risk for a star [53, *Table 2*] when moving away from the perimeter will remain true for trees as well. This fact is important: as expected, the upper bounds of the relative deviation from the optimal solution in the case of an arbitrary placement of elements in the structure at a fixed distance from the perimeter (given in [53]) can be used for trees.

To confirm this, we carry out a series of numerical experiments by analogy with [53]. Let us impose the following constraints:

$$\begin{cases} u = 1 \\ 0 < p_{M(v_{0\ldots i})}^0 \le p_{M(v_{0\ldots ij})}^0 \\ \quad \forall i \in \{1, \ldots, q_k\}, j \in \{1, \ldots, b_{ki}\}, k \in \{0, \ldots, L-1\} \\ p_{M(v_{0\ldots ij})}^0 \le \dfrac{u}{1+\sqrt{m}} \\ \quad \forall i \in \{1, \ldots, q_k\}, j \in \{1, \ldots, b_{ki}\}, k \in \{0, \ldots, L-2\} \\ p_{M(v_{0\ldots ij})}^0 \le \displaystyle\sum_{l=L}^{\infty} \left(\frac{u}{1+\sqrt{m}} \right)^{l+1} \\ \quad \forall i \in \{1, \ldots, q_{L-1}\}, j \in \{1, \ldots, b_{(L-1)i}\}. \end{cases}$$

We generate the expressions (3) for all placements obtained by permuting elements at a fixed distance k from the perimeter, starting from $k = 1$ (the first and farther layers). For each k, it is necessary to consider the cases $q_k = 2, \ldots, m$ corresponding to trees with q_k vertices of the kth layer. Then we analyze all possible absolute values for the difference of these expressions and find a global maximum for each of them. Dividing the resulting value by the minimum of the difference

of these two expressions yields the relative deviation. The maximum of such deviations is an upper bound on the relative error of the solution of problem (5).

The table represents the resulting values of the relative error for small trees.

Figure 2 shows the behavior of the relative error values at layers 1–4 depending on the number of outgoing arcs at the current layer and the number of leaves in the tree. These are the subsets $\{v_{0j}\}_{j=1}^{b_{01}}$ (Fig. 2a), $\left\{\{v_{0ij}\}_{j=1}^{b_{1i}}\right\}_{i=1}^{q_1}$ (Fig. 2b), $\left\{\{v_{0...ij}\}_{j=1}^{b_{2i}}\right\}_{i=1}^{q_2}$ (Fig. 2c), and $\left\{\{v_{0...ij}\}_{j=1}^{b_{3i}}\right\}_{i=1}^{q_3}$ (Fig. 2d). Note that it is similar to the behavior demonstrated by the values of the maximum risk increase under a given value of the marginal eigen risk. (For details, see Table 2 of the paper [53].) Namely, at the first layer the error grows with the number of leaves in the tree. At the second and farther layers, when the number of leaves is $m \geq 5$, the error decreases monotonically. Monotonicity is violated for a small number of leaves. This phenomenon was briefly described in [53]. We will not investigate this issue in more detail: the main objective is to develop risk management methods for complex network structures with thousands of vertices and edges.

Note that the experimental values of the relative deviation decrease monotonically with the distance to the perimeter. Therefore, to construct a system with an overall risk not exceeding the minimum possible one by more than 6.07% (the upper bound in Fig. 2b), it suffices to select, in an optimal way, an element for placing in the perimeter vertex and elements for placing in the first layer's vertices. Under the above condition $u_{s_i} = u \ \forall i \in \{1,...,n\}$, these are the vertices with the smallest eigen risks. Such an error is acceptable for a wide class of systems. When a higher level of protection is required, one should select q_2 additional elements from the unplaced ones with the smallest eigen risks and place arbitrarily the remaining elements in the vertices $V \setminus \left\{\{v_0\} \cup \bigcup_{j=1}^{b_{01}} \{v_{0j}\} \cup \bigcup_{i=1}^{q_1} \bigcup_{j=1}^{b_{1i}} \{v_{0ij}\}\right\}$. Then one should find the optimal placement of the selected elements in the vertices $\bigcup_{i=1}^{q_1} \bigcup_{j=1}^{b_{1i}} \{v_{0ij}\}$, e.g., by calculating $q_3!$ values of the overall risk for all possible permutations of elements at the second layer. In this case, the error of the resulting solution will be below 1.32%.

The estimated relative error of solving the problem of optimal element placement in vertex subsets of a tree structure. The values are rounded up to the fourth decimal.

The number of vertices in the subset	Vertex subset			
	$\{v_{0j}\}_{j=1}^{b_{01}}$	$\left\{\{v_{0ij}\}_{j=1}^{b_{1i}}\right\}_{i=1}^{q_1}$	$\left\{\{v_{0...ij}\}_{j=1}^{b_{2i}}\right\}_{i=1}^{q_2}$	$\left\{\{v_{0...ij}\}_{j=1}^{b_{3i}}\right\}_{i=1}^{q_3}$
Three leaves (m = 3)				
2	0.3095	0.0585	0.0119	0.0030
3	0.1548	0.0434	0.0088	0.0022
Four leaves (m = 4)				
2	0.3750	0.0571	0.0107	0.0025
3	0.2500	0.0461	0.0086	0.0020
4	0.2000	0.0607	0.0107	0.0024
Five leaves (m = 5)				
2	0.4223	0.0553	0.0097	0.0021
3	0.3168	0.0468	0.0082	0.0018
4	0.2563	0.0591	0.0098	0.0021
5	0.1709	0.0515	0.0085	0.0018
Six leaves (m = 6)				
2	0.4588	0.0535	0.0089	0.0018
3	0.3671	0.0466	0.0077	0.0016
4	0.2997	0.0574	0.0090	0.0018
5	0.2248	0.0512	0.0080	0.0016
6	0.1899	0.0607	0.0091	0.0018

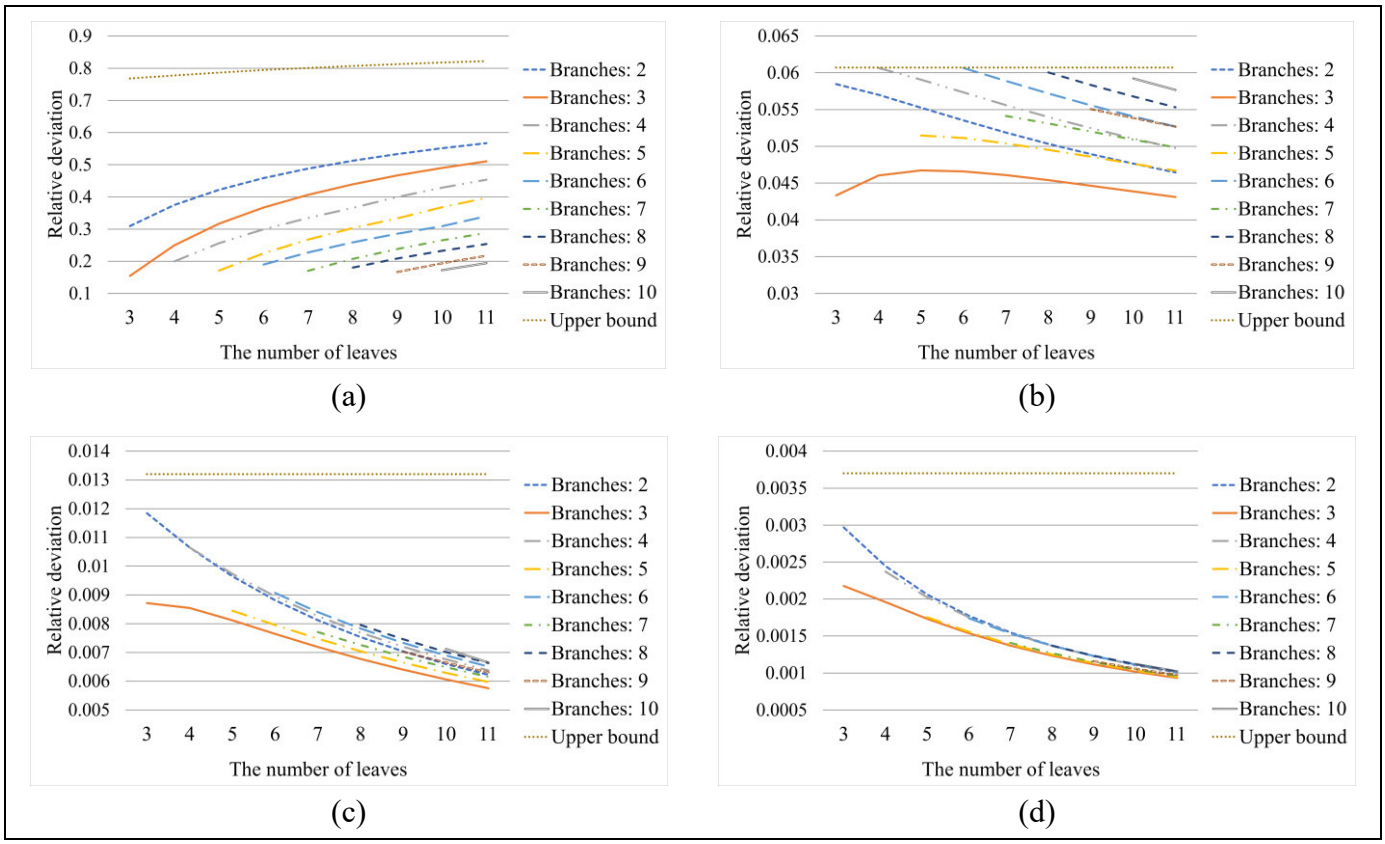

Fig. 2. The estimated relative deviation from the optimal element placement in subsets of a tree structure depending on the number of its leaves: the upper bounds in the vertex subset of the first layer (Fig. 2a) correspond to the maximum risk increment in the star structure (i.e., when the number of branches and leaves coincide); the upper bounds for the vertex subsets of layers 2–4 (Figs. 2b–2d) are the values obtained in [53] for a star structure with four rays (Fig. 2b) and two rays (Figs. 2c and 2d).

CONCLUSIONS

This paper continues a series of research works devoted to the influence of the internal structure of a complex system on its overall risk. To achieve the objective of the study, the problem of optimally placing protected system's elements in a given structure has been formulated. This problem statement allows considering the influence of the system structure on its risk regardless of the resources allocated (as in the classical Attacker–Defender problem). A direct method to solve this problem seems ambiguous, and we have decided to analyze different structures sequentially in ascending order of their complexity.

Chains have been considered in [52]. The general solution presented therein is a preference criterion to select a system element for placing in the vertex of a simple chain depending on its position to the perimeter. A star structure (one perimeter vertex and an arbitrary finite number of simple outgoing chains, particularly of infinite length) has been investigated in [53]. Upper bounds have been derived for the relative error of the problem solution under an arbitrary placement of elements starting from some distance to the perimeter.

In this paper, the upper bounds for star structures have been generalized to arbitrary trees. For this purpose, we have introduced a vertex designation system to indicate explicitly the path to the current vertex from the perimeter; have formulated the problem of optimal placement of system elements in the tree structure; and have calculated upper bounds for the relative error of the problem solution for trees with a small number of branches and leaves. Also, the behavior of these bounds has been analyzed when increasing the number of leaves and branches. According to the conclusions, the solution errors do not exceed the upper bounds obtained previously for star structures.

The results of this paper can be applied, e.g., in risk management for computer networks with variable topology, such as fog computers [55] or wireless mesh networks [56], in security system design [57], and many other fields. The approach proposed allows assessing to what extent rearranging the topology of a computer network (in another example, the structure of a security system) influences its overall protection;

the upper bounds derived allow estimating the overall risk of the system.

The next stage of research works will deal with arbitrary-topology structures with a single-vertex perimeter.

REFERENCES

1. *The Risk Management Handbook: A Practical Guide to Managing the Multiple Dimensions of Risk*, Hillson, D., Ed., London: Kogan Page Publishers, 2023.
2. ISO 31000: Risk Management – Principles and Guidelines. Geneva, Switzerland: International Organization for Standardization, 2018.
3. Rass, S., On Game-Theoretic Risk Management (Part One) – Towards a Theory of Games with Payoffs that are Probability-Distributions, *arXiv:1506.07368*, 2015. DOI: https://doi.org/10.48550/arXiv.1506.07368
4. Rass, S., On Game–Theoretic Risk Management (Part Two) – Algorithms to Compute Nash-Equilibria in Games with Distributions as Payoffs, *arXiv:1511.08591*, 2015. DOI: https://doi.org/10.48550/arXiv.1511.08591
5. Rass, S., On Game–Theoretic Risk Management (Part Three) – Modeling and Applications, *arXiv:1711.00708*, 2017. DOI: https://doi.org/10.48550/arXiv.1711.00708
6. Ostapenko, A.G., Parinov, A.V., Kalashnikov, A.O., et al., *Sotsial'nye seti i destruktivnyi kontent* (Social Networks and Destructive Content), Novikov, D.A., Ed., Moscow: Goryachaya Liniya – Telekom, 2017. (In Russian.)
7. Kalashnikov, A.O., *Modeli i metody organizatsionnogo upravleniya informatsionnymi riskami korporatsii* (Models and Methods for the Organizational Management of Corporate Information Risks), Moscow: Trapeznikov Institute of Control Sciences RAS, 2011. (In Russian.)
8. Kalashnikov, A.O. and Anikina, E.V., Management of Information Risks for Complex System Using the "Cognitive Game" Mechanism, *Cybersecurity Issues*, 2020, vol. 38, no 4, pp. 2–10. (In Russian.)
9. Deng, S., Zhang, J., Wu, D., et al., A Quantitative Risk Assessment Model for Distribution Cyber-Physical System Under Cyberattack, *IEEE Transactions on Industrial Informatics*, 2023, vol. 19, no. 3, pp. 2899–2908.
10. Hu, B., Zhou, C., Tian, Y.–C., et al., Attack Intention Oriented Dynamic Risk Propagation of Cyberattacks on Cyber-Physical Power Systems, *IEEE Transactions on Industrial Informatics*, 2023, vol. 19, no. 3, pp. 2453–2462.
11. Xiaoxiao, G., Tan, Y., and Wang, F., Modeling and Fault Propagation Analysis of Cyber-Physical Power System, *Energies*, 2020, vol. 13, no. 3, art. no. e539.
12. Gao, X., Peng, M., Tse, C.K., and Zhang, H., A Stochastic Model of Cascading Failure Dynamics in Cyber-Physical Power Systems, *IEEE Systems Journal*, 2020, vol. 14, no. 3, pp. 4626–4637.
13. Marashi, K., Sarvestani, S.S., and Hurson, A.R., Identification of Interdependencies and Prediction of Fault Propagation for Cyber-Physical Systems, *Reliability Engineering & System Safety*, 2021, vol. 215, art. no. e107787.
14. Yan, K., Liu, X., Lu, Y., and Qin, F., A Cyber-Physical Power System Risk Assessment Model Against Cyberattacks, *IEEE Systems Journal*, 2023, vol. 17, no. 2, pp. 2018–2028.
15. Pelissero, N., Laso, P.M., and Puentes, J., Impact Assessment of Anomaly Propagation in a Naval Water Distribution Cyber–Physical System, *Proceedings of 2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, Rhodes, Greece, 2021, pp. 518–523.
16. Islam, M.Z., Lin, Y., Vokkarane, V.M., and Venkataramanan, V., Cyber–Physical Cascading Failure and Resilience of Power Grid: A Comprehensive Review, *Frontiers in Energy Research*, 2023, vol. 11, art. no. e1095303.
17. Zhang, C., Xu, X., and Dui, H., Analysis of Network Cascading Failure Based on the Cluster Aggregation in Cyber-Physical Systems, *Reliability Engineering & System Safety*, 2020, vol. 202, art. no. e106963.
18. Xing, L., Cascading Failures in Internet of Things: Review and Perspectives on Reliability and Resilience, *IEEE Internet of Things Journal*, 2021, vol. 8, no. 1, pp. 44–64.
19. Wang, Q., Jia, G., Jia, Y, and Song, W., A New Approach for Risk Assessment of Failure Modes Considering Risk Interaction and Propagation Effects, *Reliability Engineering & System Safety*, 2021, vol. 216, art. no. e108044.
20. Khoshakhlagh, A., Moradi Hanifi, S., Laal, F., et al., A Model to Analyze Human and Organizational Factors Contributing to Pandemic Risk Assessment in Manufacturing Industries: FBN–HFACS Modelling, *Theoretical Issues in Ergonomics Science*, 2023, vol. 25, no. 4, pp. 369–390.
21. Moore, S. and Rogers, T., Predicting the Speed of Epidemics Spreading in Networks, *Physical Review Letters*, 2020, vol. 124, no. 6, art. no. e068301.
22. Nasution, H., Jusuf, H., Ramadhani, E., and Husein, I., Model of Spread of Infectious Diseases, *Systematic Reviews in Pharmacy*, 2020, vol. 11, no. 2, pp. 685–689.
23. Albert, R., Jeong, H., and Barabasi, A.-L., Error and Attack Tolerance of Complex Networks, *Nature*, 2000, vol. 406, pp. 378–382.
24. Artime, O., Grassia, M., De Domenico, M., et al., Robustness and Resilience of Complex Networks, *Nature Reviews Physics*, 2024, vol. 6, no. 2, pp. 114–131.
25. Ming, L., Run-Ran, L., Linyuan, L., et al., Percolation on Complex Networks: Theory and Application, *Physics Reports*, 2021, vol. 907, pp. 1–68.
26. Bak, P., Chen, K., and Tang, C., A Forest-Fire Model and Some Thoughts on Turbulence, *Physics Letters A*, 1990, vol. 147, no. 5–6, pp. 297–300.
27. Palmieri, L. and Jensen, H.J., The Forest Fire Model: The Subtleties of Criticality and Scale Invariance, *Frontiers in Physics*, 2020, vol. 8, art. no. e00257.
28. Rybski, D., Butsic, V., and Kantelhardt, J.W., Self-organized Multistability in the Forest Fire Mode, *Physical Review E*, 2021, vol. 104, no. 1, art. no. eL012201.
29. Newman, D.E., Nkei, B., Carreras, B.A., et al., Risk Assessment in Complex Interacting Infrastructure Systems, *Proceedings of 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, Big Island, HI, 2005. DOI: 10.1109/HICSS.2005.524
30. Li, X., Ji, L., Zhu, H., et al., Cellular Automata–Based Simulation of Cross-space Transmission of Energy Local Area Network Risks: A Case Study of a Power Supply Station in Beijing, *Sustainable Energy, Grids and Networks*, 2021, vol. 27, art. no. e100521.
31. Torres, M.A., Chávez-Cifuentes, J.F., and Reinoso, E., A Conceptual Flood Model Based on Cellular Automata for Probabilistic Risk Applications, *Environmental Modelling & Software*, 2022, vol. 157, art. no. e105530.
32. Sequeira, J.G.N., Nobre, T., Duarte, S., et al., Proof–of–Principle That Cellular Automata Can Be Used to Predict Infestation Risk by Reticulitermes grassei (Blattodea: Isoptera), *Forests*, 2022, vol. 13, no. 2, art. no. e237.

33.Gallos, L.K., Cohen, R., Argyrakis, P., et al., Stability and Topology of Scale-Free Networks under Attack and Defense Strategies, *Physical Review Letters*, 2005, vol. 94, no. 18, art. no. e188701.
34.Gallos, L.K., Cohen, R., Argyrakis, P., et al., Network Robustness and Fragility: Percolation on Random Graphs, *Physical Review Letters*, 2000, vol. 85, no. 25, art. no. e5468.
35.Wang, F., Dong, G., Tian, L., and Stanley, H.E., Percolation Behaviors of Finite Components on Complex Networks, *New Journal of Physics*, 2022, vol. 24, no. 4, art. no. e043027.
36.Dong, G., Luo, Y., Liu, Y., et al., Percolation Behaviors of a Network of Networks under Intentional Attack with Limited Information, *Chaos, Solitons & Fractals*, 2022, vol. 159, art. no. e112147.
37.Shao, S., Huang, X., Stanley, H.E., and Havlin, S., Percolation of Localized Attack on Complex Networks, *New Journal of Physics*, 2015, vol. 17, no. 2, art. no. e023049.
38.Dong, G., Xiao, H., Wang, F., et al., Localized Attack on Networks with Clustering, *New Journal of Physics*, 2019, vol. 21, no. 1, art. no. e013014.
39.Shang, Y., Percolation of Attack with Tunable Limited Knowledge, *Physical Review E*, 2021, vol. 103, no. 4, art. no. e042316.
40.Qing, T., Dong, G., Wang, F., et al., Phase Transition Behavior of Finite Clusters under Localized Attack, *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 2022, vol. 32, no. 2, art. no. e023105.
41.Goltsev, A.V., Dorogovtsev, S.N., and Mendes, J.F.F., K-Core (Bootstrap) Percolation on Complex Networks: Critical Phenomena and Nonlocal Effects, *Physical Review E*, 2006, vol. 73, no. 5, art. no. e056101.
42.Burleson-Lesser, K., Morone, F., Tomassone, M.S., and Makse, H.A., K-core Robustness in Ecological and Financial Networks, *Scientific Reports*, 2020, vol. 10, no. 1, art. no. 3357.
43.Shang, Y., Generalized K-cores of Networks under Attack with Limited Knowledge, *Chaos, Solitons & Fractals*, 2021, vol. 152, art. no. e111305.
44.Al Mannai, W.I. and Lewis, T.G., A General Defender-Attacker Risk Model for Networks, *The Journal of Risk Finance*, 2008, vol. 9, no. 3, pp. 244–261.
45.Peng, R., Wu, D., Sun, M., and Wu, S., An Attack-Defense Game on Interdependent Networks, *Journal of the Operational Research Society*, 2021, vol. 72, no. 10, pp. 2331–2341.
46.Ren, J., Liu, J., Dong, Y., et al., An Attacker-Defender Game Model with Constrained Strategies, *Entropy*, 2024, vol. 26, no. 8, art. no. e26080624.
47.He, S., Zhou, Y., Yang, Y., et al., Cascading Failure in Cyber-Physical Systems: A Review on Failure Modeling and Vulnerability Analysis, *IEEE Transactions on Cybernetics*, 2024, pp. 1–19. DOI: 10.1109/TCYB.2024.3411868
48.Zhou, F., Xu X., Trajcevski, G., and Zhang, K., A Survey of Information Cascade Analysis: Models, Predictions, and Recent Advances, *ACM Computing Surveys (CSUR)*, 2021, vol. 54, no. 2, pp. 1–36.
49.Cui, P., Zhu, P., Wang, K., et al., Enhancing Robustness of Interdependent Network by Adding Connectivity and Dependence Links, *Physica A*, 2018, vol. 497, pp. 185–197.
50.Xu, X. and Fu, X., Analysis on Cascading Failures of Directed-Undirected Interdependent Networks with Different Coupling Patterns, *Entropy*, 2023, vol. 25, no. 3, art. no. e471.
51.Yang, X.H., Feng, W.H., Xia, Y., et al., Improving Robustness of Interdependent Networks by Reducing Key Unbalanced Dependency Links, *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2020, vol. 67, no. 12, pp. 3187–3191.
52.Shiroky, A. and Kalashnikov, A., Mathematical Problems of Managing the Risks of Complex Systems under Targeted Attacks with Known Structures, *Mathematics*, 2021, vol. 9, no. 19, art. no. e2468.
53.Shiroky, A. and Kalashnikov, A., Influence of the Internal Structure on the Integral Risk of a Complex System on the Example of the Risk Minimization Problem in a "Star" Type Structure, *Mathematics*, 2023, vol. 11, no. 4, art. no. e998.
54.Shiroky, A.A. and Kalashnikov, A.O., Natural Computing with Application to Risk Management in Complex Systems, *Control Sciences*, 2021, no. 4, pp. 2–17. ~~(In Russian.)~~
55.Shiroky, A.A., A Method for Rapid Risk Assessment of a Fog Computing System with a Star–Shaped Topology, *Proceedings of 17th International Conference Management of Large–Scale System Development (MLSD)*, Moscow, Russia, 2024, pp. 1–5.
56.Shiroky, A., Risk Management in the Design of Computer Network Topology, in *Lecture Notes in Computer Science*, vol. 14123, Vishnevskiy, V.M., Samouylov, K.E., and Kozyrev, D.V., Eds., Cham: Springer, 2024. https://doi.org/10.1007/978-3-031-50482-2_29. (Proceedings of the 26th International Conference on Distributed Computer and Communication Networks: Control, Computation, Communications (DCCN 2023), Moscow, Russia, 2023.)
57.Shiroky, A.A., Risk Management in the Design of Security Systems with Nested Security Zones, *Proceedings of the 16th International Conference Management of Large-Scale System Development (MLSD)*, Moscow, Russia, 2023, pp. 1–4.

This paper was recommended for publication by V.N. Burkov, a member of the Editorial Board.

Received November 6, 2024, and revised March 21, 2025.
Accepted March 21, 2025.

Author information

Shiroky, Aleksandr Aleksandrovich. Cand. Sci. (Phys.–Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ shiroky@ipu.ru
ORCID ID: https://orcid.org/0000–0002–9130–5541

Kalashnikov, Andrei Olegovich. Dr. Sci. (Eng.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ aokalash@ipu.ru
ORCID ID: https://orcid.org/0000–0001–5204–1398

Cite this paper

Shiroky, A.A., Kalashnikov, A.O., How Does the Internal Structure of a Complex System Influence Its Overall Risk? Risk Minimization for Trees. *Control Sciences* **2,** 22–30 (2025).

Original Russian Text © Shiroky, A.A., Kalashnikov, A.O., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 27–37.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

A FAULT DIAGNOSIS METHOD FOR DISCRETE-EVENT SYSTEMS BASED ON THE FUZZY FINITE STATE AUTOMATON MODEL

A. E. Shumsky* and A. N. Zhirabok**

*,**Far Eastern Federal University, Vladivostok, Russia

*✉ a.e.shumsky@yandex.com, **✉ zhirabok@mail.ru

Abstract. This paper considers the problem of fault diagnosis in critical-purpose discrete-event systems described by the fuzzy finite state automaton (FSA) model. A solution method involving the mathematical apparatus of fuzzy logic is proposed. Fuzzy logic operations are described, and the concept of the determinizer of a fuzzy FSA is introduced. A diagnosis scheme that forms a structured residual vector is given. This scheme contains several channels (according to the number of possible faults in the system). Each channel is based on an observer, i.e., a determinizer of a special fuzzy FSA that simultaneously considers the possibility of both correct and incorrect transitions of the automaton (the normal operation of the system and the occurrence of a system fault, respectively). Another part of the channel is the decision block. Some ways to design the observer and the decision block are proposed. The features of the solution method are illustrated on the example of error monitoring for human operators in IT systems.

Keywords: discrete-event systems, fuzzy logic, fuzzy finite state automata, determinizer, fault diagnosis, IT systems, monitoring.

INTRODUCTION

Strict requirements for the reliability and fault tolerance of modern complex critical-purpose systems require implementing their diagnosis, i.e., the detection and isolation of faults arising during system operation, to parry or eliminate them in due time. This paper is devoted to the problem of fault diagnosis in the so-called discrete-event systems (DESs). Note that DESs include not only systems that naturally belong to this class (e.g., digital information processing and control systems).

Many systems traditionally classified as continuous (such as physical, technical (including man-machine), and socio-economic) can be treated as DESs at the top level of their hierarchy. The distinctive features of DESs are as follows [1]:

- Discrete-event systems have discrete time and discrete values of state variables.
- The state space of a DES is finite; for example, possible states are idleness, operation in a certain mode, malfunctioning, recovery, etc.
- DES operation is determined by events that can be consequences of various commands, e.g., "start operation," "change operation mode," "perform DES diagnosis," "start DES recovery," "complete DES recovery," "complete DES operation," etc.
- As a rule, a discrete-event system behaves randomly due to realizing (possibly) different transitions from one state to another initiated by the same event.

Figure 1 presents a generalized scheme of fault diagnosis and fault-tolerant control of a DES. According to this scheme, a controller generates external events for the system (DES inputs or commands). The DES responds by forming its internal events, considered to be DES outputs. The controller monitors the DES outputs and the diagnosis determined by a diagnosis system (DS) to produce a new external event (DES input). In turn, the DS monitors both external and internal events to determine the diagnosis. The latter answers the question: Is the DES in good condition? If the answer is negative, an additional judgment will be made on the type of fault. If the fault is detected and classified, the controller will report a new event, and the response will be to parry this fault (form a command sequence mitigating the fault's effect on the achievement of the system goal) or eliminate this fault (repair the DES).

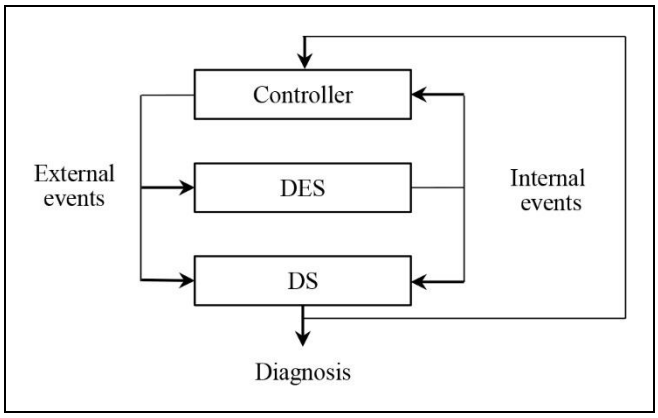

Fig. 1. The scheme of fault diagnosis and fault-tolerant control of DESs.

A fairly complete review of the existing DES diagnosis methods can be found in the papers [2, 3] and monograph [4]. As noted in [3], the following mathematical models are most widespread to describe DESs:

– deterministic finite state automata (FSA),

– probabilistic FSA and Markov chains,

– Petri nets.

This paper focuses exclusively on the application of FSA models. With such models used to diagnose DESs, the inputs and outputs of an FSA are formed as observable events (external and internal, respectively), while the occurrence of a DES fault is treated as a directly unobservable internal event.

Whenever no deterministic FSA model of a DES is available, in addition to probabilistic FSA, nondeterministic and fuzzy FSA can be used. As compared to probabilistic FSA models, nondeterministic and fuzzy models allow considerably reducing the volume of necessary calculations, thereby accelerating the fault diagnosis process. Nondeterministic and fuzzy FSA models cover the situation when transitions to different states can be realized for a fixed state and fixed input of an automaton. For nondeterministic FSA, it is impossible to give priority to the realization of a certain transition; for fuzzy FSA, however, additional (e.g., statistical) information, expert assessments, training results, etc. can be utilized to talk about the degree of confidence in the realization of each possible transition.

For diagnosing DESs described by the nondeterministic FSA model, methods based on pairwise partition algebra and pairwise covering algebra were presented in [5] and [6], respectively; also, see [7].

The objective of this paper is to develop a new fault diagnosis method based on the fuzzy FSA model. We involve mathematical constructs of fuzzy logic [8] as well as the concepts of a fuzzy finite state automaton [9] and its determinizer [10].

The features of this method are illustrated on the example of error monitoring for human operators in IT systems; it was previously considered in [5] and [6] for fault diagnosis within the nondeterministic FSA model.

1. THE OBJECTIVES AND STRUCTURE OF THIS PAPER

1.1. Models Used

Let a DES in good condition be described by the fuzzy FSA model

$$A=\left(U,X,Y,\delta,\lambda,x(0)\right), \tag{1}$$

where $U=\{u_1, u_2, \ldots, u_m\}$, $X=\{x_1, x_2, \ldots, x_n\}$, and $Y=\{y_1, y_2, \ldots, y_l\}$ denote the finite sets of inputs, states, and outputs, respectively; $x(0)\in X$ is a known initial state; $\delta: X\times U\to\mu(X)$ is a fuzzy transition function; $\mu(X)\in\{\mu(x_i)\in[0,1],\ 1\le i\le n\}$ is a fuzzy set; finally, $\lambda: X\to Y$ is an output function.

We utilize the matrix representation for the fuzzy transition function δ: the transitions performed under an input u_k are described by a matrix M^k of dimensions $n\times n$, in which each element $M^k_{i,j}\in[0,1]$ characterizes the degree of confidence that, given $u_k\in U$, an automaton A will move from a state $x_i\in X$ to a state $x_j\in X$. Let $S(U)$ denote the set of all matrices $M^k, 1\le k\le m$.

If it is impossible to specify the degree of confidence in transitions reasonably, we propose to proceed as follows: set an element of the matrix $M^k_{i,j}$ corresponding to an admissible transition equal to 1 whereas the element of this matrix corresponding to an inadmissible transition equal to 0. Thereby, one passes from the fuzzy FSA model of a DES to its nondeterministic counterpart.

We specify the output function λ using a matrix L of dimensions $l\times n$ in which $L_{i,j}=1$ if the output $y_i\in Y$ is generated by the automaton A in the state $x_j\in X$ and $L_{i,j}=0$ otherwise. Let $S(Y)$ denote the set of all rows $\{L_i, 1\le i\le l\}$ of the matrix L.

Consider $X_{i,k}$, $X_{i,k}\subseteq X$, the set of all states reachable from a state $x_i\in X$ under an input $u_k\in U$. Assume that a fault $f_s, 1\le s\le N$, in the DES model (1) can be represented by a distortion of the transition

function δ such that, given $u_k \in U$, an inadmissible transition from a state x_i to a state $x_t \notin X_{i,k}$ is realized instead of an admissible transition from the former state to a state $x_j \in X_{i,k}$. This fact will be indicated by $f_s : \left(x_j \to x_t\right)_{i,k}$. In this case, the matrix M^k of the faulty DES will be obtained by changing the value of the element $M^k_{i,t}$ of the matrix M^k of the operable DES from 0 to 1. To simplify the presentation, we accept the hypothesis of single faults from a predetermined list $F = \{f_1, f_2 \ldots, f_N\}$. Let A_s denote an auxiliary automaton whose transition function δ_s is obtained by changing (in the above manner) the transition function δ of the automaton A to the transition caused by the fault f_s:

$$A_s = \left(U, X, Y, \delta_s, \lambda, x(0)\right). \qquad (2)$$

Due to the construction procedure of the matrix M^k, model (2) covers the possibility of a "correct" transition in the operable DES and, moreover, the possibility of an "incorrect" transition to the state caused by the DES fault f_s. Therefore, there is a definite correspondence between the behavior of model (2) and the behavior of both the operable and faulty DESs. In subsection 3.2, we propose a method for evaluating this correspondence in terms of possibility (confidence) and form a structured residual vector based on the method.

1.2. Fault Diagnosis Scheme

To detect and isolate faults, the idea is to use the fault diagnosis scheme shown in Fig. 2. This scheme contains N channels (according to the number of possible DES faults), and each channel includes a deterministic FSA A^d_s and a decision block DB_s. For the diagnosis scheme design, it is necessary to determine the components of each channel. Therefore, the remainder of this paper is organized as follows.

1.3. The Structure of This Paper

Section 2 provides the mathematical constructs of fuzzy logic required to obtain the main results of the paper. In Section 3, the design problem of a deterministic FSA A^d_s (called the observer of the nondeterministic FSA A_s) is reduced to the modified problem of finding the determinizer [10] of this automaton. Also, we propose an operation rule for DB_s, $1 \le s \le N$, ensuring the structuredness of the residual vector, an important property with the following essence. First, the zero value of all the components of the residual vector means the absence of DES faults. Second, if only one component of the residual vector is 0 (the others being equal to 1), then the DES has a fault with the number coinciding with that of the zero component of the residual vector. In Section 4, we consider a numerical example and simulation results to illustrate the features of the method proposed. The outcomes of this paper are summarized in the Conclusions.

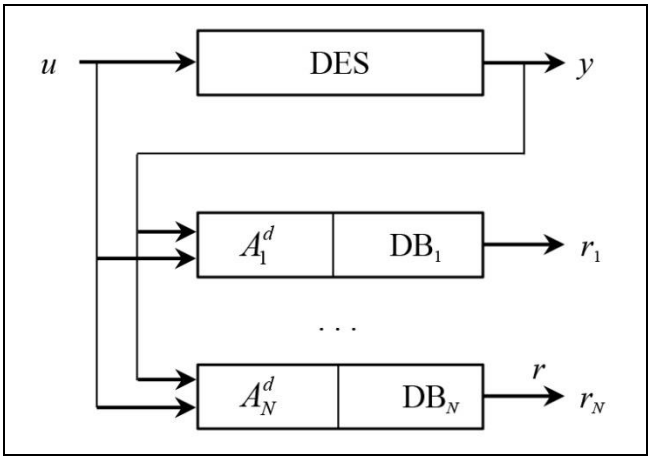

Fig. 2. The fault diagnosis scheme of DESs.

2. MATHEMATICAL CONSTRUCTS

2.1. Fuzzy Logic Operations

Recall several operations of fuzzy logic [8], playing an important role for the further presentation. A fuzzy matrix $B = \{B_{ij}\}$ is a matrix with elements $B_{ij} \in [0,1]$. Hence, the above matrices M^k, $1 \le k \le m$, are fuzzy. Now let B and C be fuzzy matrices of dimensions $a \times b$ and $b \times c$, respectively. The product of fuzzy matrices is defined by [10]

$$\left(BC\right)_{ij} = \max_{i,j} \min \left(B_{ih}\ C_{hj}\right),$$

where B_{ih} and C_{hj} are the corresponding elements of matrices B and C, $1 \le i \le a$, $1 \le h \le b$, and $1 \le j \le c$. This formula generalizes the well-known matrix multiplication [11, *p. 24*]; it is obtained by replacing the product of matrix elements by the operation of finding the minimum and the sum of elements by the operation of finding the maximum. Here is a simple numerical example.

Example 1. Consider matrices B and C of the form

$$B = \begin{pmatrix} 0.1 & 0.9 \\ 0.7 & 0.2 \end{pmatrix},\ C = \begin{pmatrix} 0.6 & 0.3 \\ 0.4 & 0.8 \end{pmatrix}.$$

The calculations yield the following results: $(BC)_{11} = \max\left(\min(0.1 \;\; 0.6), \min(0.9 \;\; 0.4)\right) = 0.4$, $(BC)_{12} = \max\left(\min(0.1 \;\; 0.3), \min(0.9 \;\; 0.8)\right) = 0.8$, $(BC)_{21} = \max\left(\min(0.7 \;\; 0.6), \min(0.2 \;\; 0.4)\right) = 0.6$, $(BC)_{22} = \max\left(\min(0.7 \;\; 0.3), \min(0.2 \;\; 0.8)\right) = 0.3$. Thus, the product of the fuzzy matrices B and C is

$$BC = \begin{pmatrix} 0.4 & 0.8 \\ 0.6 & 0.3 \end{pmatrix}. \blacklozenge$$

2.2. The Concept of the Determinizer of a Fuzzy FSA

The definition of the determinizer of a fuzzy FSA [10] is introduced as follows. Let E be some set of fuzzy n-dimensional row vectors whose components can take values on the interval $[0, 1]$. Also, let E^0, $E^0 \subseteq E$, be the set of all unit n-dimensional row vectors, called the generating states of the determinizer $D(A')$. The closure $[E]_\Sigma$ of the set E with respect to a signature (set of admissible operations) Σ is the set of all vectors, including those from E, that can be obtained by applying operations from the signature Σ to vectors from E. We construct the closure using the following procedure.

Step 1. Assign $i = 0$.

Step 2. Find the vector set $E^{i+1} = E^i \cup \left[E^i\right]_\Sigma$.

Step 3. If $E^{i+1} = E^i$, assign $[E]_\Sigma = E^i$ and complete the procedure.

Step 4. Otherwise, let $i = i + 1$ and go back to Step 2.

Let $A' = (U, X, \delta)$ be a fuzzy semiautomaton (i.e., an automaton A without output function) and $S(U) = \{M^k, 1 \le k \le m\}$ be the set of fuzzy transition matrices of the automaton A. Assume that the signature Σ includes all operations involving the multiplication of an n-dimensional row vector on the right by a matrix from the set $S(U)$. The determinizer of a fuzzy semiautomaton A' is a deterministic FSA described by the triple

$$D(A') = (S(X), U, \Delta),$$

where $S(X) = [E]_{S(U)}$ and $\Delta : S(X) \times S(U) \to S(X)$ is the determinizer's transition function defined by

$$\Delta(\mu_i, u_k) = \mu_i \cdot M^k, \; \mu_i \in S(X), \; M^k \in S(U). \quad (3)$$

To explain formula (3), we recall that the matrix M^k describes a transition activated by an input u_k.

Example 2. Consider a semiautomaton A' defined by the fuzzy transition matrices

$$M^1 = \begin{pmatrix} 0.4 & 0.6 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \; M^2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

Let the generating states of the determinizer be $\mu_1 = (1 \;\; 0 \;\; 0)$, $\mu_2 = (0 \;\; 1 \;\; 0)$, and $\mu_3 = (0 \;\; 0 \;\; 1)$. The necessary calculations by formula (3) yield the following vectors μ_4, μ_5, and μ_6 of the closure $[E]_{S(U)}$: $\mu_4 = \mu_1 \cdot M^1 = (0.4 \;\; 0.6 \;\; 0)$, $\mu_5 = \mu_4 \cdot M^1 = (0.4 \;\; 0.4 \;\; 0.6)$, $\mu_6 = \mu_4 \cdot M^2 = (0.6 \;\; 0 \;\; 0.4)$. Following similar considerations, we obtain $\mu_7 = (0.4 \;\; 0.6 \;\; 0.4)$, $\mu_8 = (0.6 \;\; 0.4 \;\; 0.4)$, $\mu_9 = (0 \;\; 0.4 \;\; 0.6)$, $\mu_{10} = (0 \;\; 0 \;\; 0.4)$, $\mu_{11} = (0 \;\; 0.4 \;\; 0)$, $\mu_{12} = (0.4 \;\; 0 \;\; 0)$, $\mu_{13} = (0.4 \;\; 0.4 \;\; 0)$, $\mu_{14} = (0.4 \;\; 0.4 \;\; 0.4)$, $\mu_{15} = (0.4 \;\; 0 \;\; 0.4)$, $\mu_{16} = (0 \;\; 0.4 \;\; 0.4)$.

Finally, taking $M(X) = \{\mu_i, 1 \le i \le 16\}$, we construct the transition table of the determinizer $D(A')$ (Table 1). For example, it indicates the transitions from the state μ_9 to the state μ_{10} under the input u_1 and those from the state μ_9 to the state μ_4 under the input u_2 since $\mu_9 \cdot M^1 = \mu_{10}$ and $\mu_9 \cdot M^2 = \mu_4$. Note that the number of states of the determinizer $D(A')$ significantly exceeds that of the original semiautomaton A'. Indeed, when constructing the determinizer, we preserve all information about the degree of confidence in realizing each possible transition since each determinizer's state is a vector of possible DES transitions to the corresponding states at a definite time instant. ♦

Table 1

The transition table of the determinizer $D(A')$

	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8	μ_9	μ_{10}	μ_{11}	μ_{12}	μ_{13}	μ_{14}	μ_{15}	μ_{16}
u_1	μ_4	μ_3	μ_3	μ_5	μ_5	μ_7	μ_5	μ_7	μ_{10}	μ_{10}	μ_{10}	μ_{13}	μ_{14}	μ_{14}	μ_{14}	μ_{10}
u_2	μ_3	μ_1	μ_2	μ_6	μ_7	μ_9	μ_8	μ_5	μ_4	μ_{11}	μ_{12}	μ_{10}	μ_{15}	μ_{14}	μ_{16}	μ_{13}

To obtain an upper bound for the number of determinizer's states, we consider that the components of all vectors from the set $S(X)$ can take only the values contained in the matrices from the set $S(U)$. Excluding the zero vector from the analysis gives the formula

$$\#S(X) \le q^n - 1, \tag{4}$$

where # denotes the set cardinality; q is the number of different values for the elements of fuzzy matrices from the set $S(U)$; as before, n stands for the number of states of the fuzzy FSA. For example, consider system (3); in this case, we have $q = 4$ (the set of different values for the elements of the matrices M^1 and M^2 includes the numbers $\{0;\ 0.4;\ 0.6;\ 1\}$), $n = 3$, and, consequently, $\#S(X) \le 4^3 - 1 = 63$.

Example 3. The above approach can be applied to the determinization of nondeterministic FSA. As an illustration, we consider the nondeterministic FSA model obtained from the fuzzy one (3) by setting the unit degrees of confidence for all admissible transitions:

$$M^1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, M^2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}. \tag{5}$$

In the case of nondeterministic FSA, the dimension of the resulting determinizer is not necessarily smaller than that of the original automaton. Indeed, for nondeterministic FSA we have $q = 2$ and, by formula (4), the number of determinizer's states is

$$\#S(X) \le 2^n - 1.$$

Particularly for the nondeterministic FSA (5), it follows that $\#S(X) \le 2^3 - 1 = 7$. Omitting intermediate calculations, we directly present the transition table of the determinizer of nondeterministic FSA (5).

In Table 2, the determinizer's states are $\mu_1 = (1\ \ 0\ \ 0)$, $\mu_2 = (0\ \ 1\ \ 0)$, $\mu_3 = (0\ \ 0\ \ 1)$, $\mu_4 = (1\ \ 1\ \ 0)$, $\mu_5 = (1\ \ 1\ \ 1)$, $\mu_6 = (1\ \ 0\ \ 1)$, and $\mu_7 = (0\ \ 1\ \ 1)$. ♦

Table 2

The transition table of the determinizer of the nondeterministic FSA (5)

	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7
u_1	μ_4	μ_3	μ_3	μ_5	μ_5	μ_5	μ_3
u_2	μ_3	μ_1	μ_2	μ_6	μ_5	μ_7	μ_4

Note that the algebraic determinization approach for nondeterministic FSA based on partition algebra [5] and covering algebra [6] surely yields a deterministic FSA with a dimension not exceeding that of the original automaton. The reason is that the algebraic approach does not require preserving information about the degree of confidence in the realization of each possible transition. The fuzzy approach under consideration involves additional useful information about the degree of confidence in the realization of each possible transition during the fault diagnosis procedure, providing a potential opportunity to increase (if necessary) the depth of fault isolation. The price for this is a significant dimension of the determinizer's transition table.

The main difference between the descriptions of the deterministic FSA A_s^d, $1 \le s \le N$, (the fault diagnosis scheme in Fig. 2) and the determinizer $D(A_s')$ of the corresponding semiautomaton A_s' is that the transition function of the automaton A_s^d additionally depends on the outputs of the original automaton. As a result, additional information can be used to adjust the behavior and reduce the dimension of the automaton A_s^d. Drawing an analogy with the observers of a continuous dynamic system, we call the deterministic FSA A_s^d an observer of the fuzzy FSA A_s.

3. DIAGNOSIS CHANNEL DESIGN

3.1. The Observer A_s^d

The further presentation concerns the channel of the fault diagnosis scheme intended to isolate a fault f_s. First, consider a method for finding the transition function δ_s^d of the observer A_s^d based on a slight modification of the relation (3). For vectors $\mu_i \in S(X)$ and $L_j \in S(Y)$, we introduce the notation

$$\langle \mu_i,\ L_j \rangle = \big(\min(\mu_{i,1},\ L_{j,1}),\ \min(\mu_{i,2},\ L_{j,2}),\ldots,\ \min(\mu_{i,n},\ L_{j,n})\big). \tag{6}$$

Let the transition function $\Delta : S(X) \times S(Y) \times S(U) \to S(X)$ of the deterministic FSA A_s^d be defined as follows:

$$\Delta(\mu_i,\ y_j,\ u_k) = \langle \mu_i,\ L_j \rangle \cdot M^k, \quad \mu_i \in S(X),\ L_j \in S(Y),\ M^k \in S(U). \tag{7}$$

With $\langle \mu_i, L_j \rangle$ used in formula (7) instead of μ_i in the determinizer's transition function (3), we specify the values of membership functions (to the corresponding fuzzy sets) for the components of the vector $x \in X$ before the transition.

Example 4. Consider a fuzzy FSA described by the matrices M^1 and M^2 of Example 2 (in the absence of faults). Let the output function of this automaton be given by the matrix

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}.$$

Its transition errors are a consequence of two faults, $f_1 : (x_3 \to x_1)_{1,2}$ and $f_2 : (x_3 \to x_2)_{1,2}$. The automaton's matrices M_1^2 and M_2^2 including an additional erroneous transition due to an appropriate fault (indicated by the matrix subscript) have the form

$$M_1^2 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \; M_2^2 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

The matrix M^1 is unaffected by the fault and retains its original form.

Now we design the observer A_1^d (in this case, using only the matrix M_1^2). As in the previous example, let the generating states be $\mu_1 = (1 \;\; 0 \;\; 0)$, $\mu_2 = (0 \;\; 1 \;\; 0)$, and $\mu_3 = (0 \;\; 0 \;\; 1)$. Calculations according to the right-hand side of the relation (7) give

$$\langle \mu_1, L_1 \rangle \cdot M^1 = \begin{pmatrix} \min(1,1) & \min(0,0) & \min(0,0) \end{pmatrix} \times \begin{pmatrix} 0.4 & 0.6 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} = (0.4 \;\; 0.6 \;\; 0) = \mu_4.$$

By analogy, we find $\mu_5 = (1 \;\; 0 \;\; 1)$, $\mu_6 = (0.4 \;\; 0.4 \;\; 0)$, $\mu_7 = (0 \;\; 0 \;\; 0.6)$, $\mu_8 = (0.4 \;\; 0 \;\; 0.4)$, $\mu_9 = (0.6 \;\; 0 \;\; 0)$, $\mu_{10} = (0 \;\; 0 \;\; 0.4)$, $\mu_{11} = (0.4 \;\; 0 \;\; 0)$, $\mu_{12} = (0 \;\; 0.6 \;\; 0)$, $\mu_{13} = (0 \;\; 0.4 \;\; 0)$, and $\mu_{14} = (0.6 \;\; 0 \;\; 0.6)$ and build the transition table of the observer A_1^d (Table 3).

In this table, dashes indicate the transitions corresponding to the incompatible values of the observer state and DES output. For example, the observer state μ_2 allows only the DES output y_2, which is immediate from the expressions for the vector μ_2 and the matrix L. Hence, the combination of μ_2 and y_1 is impossible during the faultless operation of the DES.

Finally, we compare the dimension of the determinizer $D(A')$ (Table 1) and the dimension of the observer A_1^d (Table 3), emphasizing that the latter does not exceed the former. However, in practice, the observer may have a higher dimension than the original fuzzy FSA. ♦

Seemingly, this fact should limit the practical realizability of the fault diagnosis procedure of DESs based on the fuzzy FSA model due to the significant dimension of real systems. As expected, these limitations should be less pronounced within the algebraic approach [5, 6]. Meanwhile, the actual things differ. The point is that the algebraic approach finds a tabular description for the transition function of observers (deterministic FSA A_i^d, $1 \le i \le N$) from the tabular description of the original finite state automaton model of the DES. In contrast, the approach proposed specifies the transition function in a compact analytical form, being therefore insensitive to the growth of the dimension of the original DES model. Moreover, with this feature, the approach proposed is preferable when designing fault diagnosis schemes for DESs based on the deterministic and nondeterministic FSA models. It successfully overcomes the so-called "curse of dimensionality," inevitably arising for all methods with a tabular (or graph-based) description of the FSA model of DESs.

3.2. The Decision Block of the Diagnosis Channel

Assume that during the fault diagnosis procedure, the DES output y_j and the state μ_i of the observer A_s^d are generated simultaneously. Let σ_s denote the value of the maximum component of the vector $\langle \mu_i, L_j \rangle$. This value will be regarded as the degree of

Table 3

The transition table of the observer A_1^d

		μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	μ_8	μ_9	μ_{10}	μ_{11}	μ_{12}	μ_{13}	μ_{14}
u_1	y_1	μ_4	-	-	μ_6	μ_4	μ_6	-	μ_6	μ_4	-	μ_6	-	-	μ_4
	y_2	-	μ_3	μ_3	μ_7	μ_3	μ_{10}	μ_7	μ_{10}	-	μ_{10}	-	μ_7	μ_{10}	μ_7
u_2	y_1	μ_5	-	-	μ_8	μ_5	μ_8	-	μ_8	μ_{14}	-	μ_8	-	-	μ_{14}
	y_2	-	μ_1	μ_2	μ_{12}	μ_2	μ_{11}	μ_{12}	μ_{13}	-	μ_{13}	-	μ_{11}	μ_{13}	μ_{12}

confidence that the DES behavior corresponds to the behavior of the observer A_s^d. In view of the aforesaid, we specify the relation Ψ_s:

$$\left(\mu_i,\, y_j\right) \in \Psi_s \Leftrightarrow \sigma_s \neq 0. \qquad (8)$$

Assume now that the relation Ψ_s (8) has been verified as false for a particular pair y_j and μ_i (i.e., $\sigma_s = 0$). This indicates the presence of an error (fault) f_k, $k \neq s$, in the DES. In this case, the corresponding residual value is $r_s = 1$; otherwise $(\sigma_s \neq 0)$, $r_s = 0$. To judge unambiguously the fault type in the DES based on the structured residual vector, only one of its components should retain the zero value. (The number of this component will be the fault number.) If not (several residuals take zero values), then extra checks are required, e.g., using additional measurements and/or special tests [12]. Such checks should be carried out sequentially for all faults in the order of decreasing degrees of confidence from the list $\{\sigma_1, \sigma_2, ..., \sigma_N\}$.

4. AN ILLUSTRATIVE EXAMPLE

The change management process in an IT system was described in detail in [5, *Fig. 1*; 6], including a method for obtaining its deterministic and nondeterministic FSA models. Note that this process is one of the most important for IT systems: it is responsible for managing the lifecycle of all changes and facilitates the implementation of useful changes with minimum interruption of IT services. The change management process involves the following participants:

– the initiator (an IT department representative who performs the initial processing, assignment, and control of changes);

– the executor (an engineer who makes changes in configuration elements or coordinates the contractor's work on these changes);

– the Advisory Change Committee (ACC), an advisory body that meets regularly to assess and plan changes);

– the process manager (an IT department representative who controls the change management process and forms suggestions for its improvement).

The transition and output tables of the deterministic and nondeterministic FSA models of the process were also presented in the papers cited. The deterministic FSA model describes the actions of the process participants in full compliance with the regulations prescribed. In practice, a nondeterministic FSA should be used as the initial model due to some (non-critical) deviations of the participants' actions from the prescribed regulations, allowed to clarify the regulations or the participants' knowledge of them. In particular, the following situations were considered:

– The process manager sends the reviewed result of a completed task to the ACC for re-approval.

– The process manager sends a received and agreed task back to the initiator.

– The executor sends a received and agreed task back to the initiator.

– The executor sends a received and agreed task back to the ACC for approval.

The following external events are considered to be model inputs: u_1 (work plan completion), u_2 (plan approval by the ACC), u_3 (transfer of the non-approved plan for revision), u_4 (transfer of a task and a work plan to the executor), u_5 (work completion), and u_6 (entering of the changes made in the IT system library). The following stages of the regulations are considered to be model states: x_1 (formation of a task and an implementation plan), x_2 (coordination of a task and a work plan by the ACC), x_3 (coordination of a task and a work plan by the process manager), x_4 (carrying out works by the executor), x_5 (check of the carried out works by the process manager), and x_6 (completed works). Available information about some stages of the regulations is used as outputs (see the output function below).

The fuzzy FSA model of the change management process in an IT system that describes the error-free work of all process participants (the initiator, executor, and manager) can be obtained based on the nondeterministic counterpart [6, *Table 3*] and the available statistical information about the possible actions of the participants when following the regulations:

$$M^1 = \begin{pmatrix} 0 & 1 & 0 & & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

$$M^2 = \begin{pmatrix} 0 & 0 & & 0 & 0 & & 0 \\ 0 & 0 & & 1 & 0 & \cdots & 0 \\ 0 & 0 & \vdots & 0 & 0 & & 0 \\ 0 & 0 & & 0 & 0 & \ddots & \vdots \\ 0 & 0 & & 0 & 0 & \cdots & 0 \end{pmatrix},$$

$$M^3 = \begin{pmatrix} 0 & 0 & 0 & & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

$$M^4 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1 & 0.1 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$M^5 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0.1 & 0 & 0 & 0.7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$M^6 = \begin{pmatrix} 0 & & 0 & \cdots & 0 & & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & & 0 & \cdots & 0 & & 1 \\ 0 & & 0 & & 0 & & 0 \end{pmatrix}.$$

All these matrices have dimensions 6×6. The output function is described by the matrix

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

As in [6], we consider the errors of the initiator, executor, and process manager: $f_1 : (x_2 \to x_3)_{(x_1, u_1)}$, $f_2 : (x_5 \to x_6)_{(x_4, u_5)}$, and $f_3 : (x_3 \to x_4)_{(x_2, u_2)}$, respectively. The corresponding matrices are

$$M_1^1 = \begin{pmatrix} 0 & 1 & 1 & & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

$$M_2^5 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0.1 & 0 & 0 & 0.7 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$M_3^2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

All other matrices retain their values.

The tables below present the simulations for different admissible results of the participants to the change management process as well as the errors of the initiator, executor, and process manager. The residuals and auxiliary variables were generated using the relations (6)-(8) and the above matrices. The matrices used are: M_1^1, M^2, M^3, M^4, M^5, and M^6 (the first channel); M^1, M^2, M^3, M^4, M_2^5, and M^6 (the second channel); M^1, M_3^2, M^4, M^5, and M^6 (the third channel). The following scenarios were simulated.

Scenario 1. The process regulations are implemented without any deviations (Table 4).

Scenario 2. The process regulations are implemented with an admissible deviation due to the manager's transfer of task formation and work plan for re-approval by the ACC (Table 5).

Scenario 3. The process regulations are implemented with errors made by the initiator, executor, and process manager, respectively (Table 6).

Table 4

Scenario 1

Characteristics		Initial value	Input				
			u_1	u_2	u_4	u_5	u_6
System	State x	x_1	x_2	x_3	x_4	x_5	x_6
	Output y	y_1	y_2	y_3	y_4	y_2	y_5
Channel 1	State μ	μ_{10}	μ_{11}	μ_{12}	μ_{14}	μ_{15}	μ_{16}
	Confidence σ_1	1	1	1	0.8	0.7	0.7
	Residual r_1	0	0	0	0	0	0
Channel 2	State μ	μ_{20}	μ_{21}	μ_{22}	μ_{24}	μ_{25}	μ_{26}
	Confidence σ_2	1	1	1	0.8	0.7	0.7
	Residual r_2	0	0	0	0	0	0
Channel 3	State μ	μ_{30}	μ_{31}	μ_{32}	μ_{34}	μ_{35}	μ_{36}
	Confidence σ_3	1	1	1	0.8	0.7	0.7
	Residual r_3	0	0	0	0	0	0

Table 5

Scenario 2

Characteristics		Initial value	Input						
			u_1	u_2	u_4	u_2	u_4	u_5	u_6
System	State x	x_1	x_2	x_3	x_2	x_3	x_4	x_5	x_6
	Output y	y_1	y_2	y_3	y_2	y_3	y_4	y_2	y_5
Channel 1	State μ	μ_{10}	μ_{11}	μ_{12}	μ_{14}	μ_{12}^*	μ_{14}^*	μ_{15}^*	μ_{16}^*
	Confidence σ_1	1	1	1	0.1	0.1	0.1	0.1	0.1
	Residual r_1	0	0	0	0	0	0	0	0
Channel 2	State μ	μ_{20}	μ_{21}	μ_{22}	μ_{24}	μ_{22}^*	μ_{24}^*	μ_{25}^*	μ_{26}^*
	Confidence σ_2	1	1	1	0.1	0.1	0.1	0.1	0.1
	Residual r_2	0	0	0	0	0	0	0	0
Channel 3	State μ	μ_{30}	μ_{31}	μ_{32}	μ_{34}	μ_{32}^*	μ_{34}^*	μ_{35}^*	μ_{36}^*
	Confidence σ_3	1	1	1	0.1	0.1	0.1	0.1	0.1
	Residual r_3	0	0	0	0	0	0	0	0

Table 6

Scenario 3

Characteristics		Initial value	f_1	f_2				f_3	
			Input						
			u_1	u_1	u_2	u_4	u_5	u_1	u_2
System	State x	x_1	x_3	x_2	x_3	x_4	x_6	x_2	x_4
	Output y	y_1	y_3	y_2	y_3	y_4	y_5	y_2	y_4
Channel 1	State μ	μ_{10}	μ_{11}	μ_{11}	μ_{12}	μ_{14}	μ_{15}	μ_{11}	μ_{12}
	Confidence σ_1	1	1	1	1	0.8	0	1	0
	Residual r_1	0	0	0	0	0	1	0	1
Channel 2	State μ	μ_{20}	μ_{21}	μ_{21}	μ_{22}	μ_{24}	μ_{25}	μ_{21}	μ_{22}
	Confidence σ_2	1	0	1	1	0.8	0.8	1	0
	Residual r_2	0	1	0	0	0	0	0	1
Channel 3	State μ	μ_{30}	μ_{31}	μ_{31}	μ_{32}	μ_{34}	μ_{35}	μ_{31}	μ_{32}
	Confidence σ_3	1	0	1	1	0.8	0	1	1
	Residual r_3	0	1	0	0	0	1	0	0

Note that in different scenarios, each stage of the regulations is implemented a different number of times; this fact is reflected in the tables. The states of the observers of fault diagnosis channels appearing in the tables have the following values: $\mu_{10} = \mu_{20} = \mu_{30} = (1\ 0\ 0\ 0\ 0\ 0)$, $\mu_{11} = (0\ 1\ 1\ 0\ 0\ 0)$, $\mu_{21} = \mu_{31} = (0\ 1\ 0\ 0\ 0\ 0)$, $\mu_{12} = \mu_{22} = (0\ 0\ 1\ 0\ 0\ 0)$, $\mu_{32} = (0\ 0\ 1\ 1\ 0\ 0)$, $\mu_{14} = \mu_{24} = \mu_{34} = (0.1\ 0.1\ 0\ 0.8\ 0\ 0)$, $\mu_{15} = \mu_{35} = (0.2\ 0.1\ 0\ 0\ 0.7\ 0)$, $\mu_{25} = (0.2\ 0.1\ 0\ 0\ 0.7\ 0.8)$, $\mu_{16} = \mu_{26} = \mu_{36} = (0\ 0\ 0\ 0\ 0\ 0.7)$. The states of the observers of fault diagnosis channels appearing in Table 5 have the following val-

ues: $\mu^*_{12} = \mu^*_{22} = (0\ 0\ 0.1\ 0\ 0\ 0)$, $\mu^*_{32} = (0\ 0\ 0.1\ 0.1\ 0\ 0)$, $\mu^*_{14} = \mu^*_{24} = \mu^*_{34} = (0.1\ 0.1\ 0\ 0.1\ 0\ 0)$, $\mu^*_{15} = \mu^*_{35} = (0.1\ 0.1\ 0\ 0\ 0.1\ 0)$, $\mu^*_{25} = (0.1\ 0.1\ 0\ 0\ 0.1\ 0.1)$, $\mu^*_{16} = \mu^*_{26} = \mu^*_{36} = (0\ 0\ 0\ 0\ 0\ 0.1)$. The subscripts of the states are interpreted as follows: the first number corresponds to the channel number whereas the second to the input number affecting the transition.

For the sake of convenience, the superscript * indicates the newly appearing states in scenario 2 that are generated by the same inputs as in scenarios 1 and 3. This is due to implementing more steps of the regulations in scenario 2.

According to Tables 4 and 5, both under the full compliance with the regulations and an admissible deviation from the regulations caused by the process manager's action, zero values of the residuals are formed at the channel outputs. At the same time (see Table 6), when errors occur in the actions of the initiator, executor, and process manager, the resulting structured residual vector allows unambiguously concluding on the error type at the time of its occurrence.

CONCLUSIONS

This paper has proposed a fault diagnosis method for DESs described by the fuzzy FSA model. In comparison with the nondeterministic counterpart, this model can contribute to achieving the required depth of fault diagnosis. Indeed, let the generated residual vector yield no unambiguous conclusion on the fault type. (In other words, the vector contains several zero components.) In this case, several additional checks may be required to localize the fault. To reduce the number of such checks, they should be performed in the order of decreasing the degree of confidence σ_s, $1 \le s \le N$.

Obviously, the method can be extended to simpler models in the form of deterministic and nondeterministic finite state automata. Distinctive features of the method are as follows: no preliminary tabular description of diagnosis means is required; all calculations are carried out directly during the fault diagnosis process using compact analytical relations. This allows overcoming the "curse of dimensionality," which inevitably arises for the methods with the tabular (or graph-based) description of the FSA model of DESs. Thus, the earlier existing limit on the admissible dimension of the model of the DES diagnosed is almost eliminated. The method proposed can be further developed for the diagnosability analysis and verification of DES models with a large number of states [13].

Acknowledgments. *This work was supported by the Ministry of Science and Higher Education of the Russian Federation, project no. FZNS-2023-0011.*

REFERENCES

1. Wonham, W.M., Cai, K., and Rudle, K., Supervisory Control of Discrete-Event Systems: A Brief History 1980–2015, *Proceedings of the 20th IFAC Congress*, Toulouse, France, 2017, pp. 1827–1833.
2. Sampath, M., Sengupta, R., Lafortune, S., et al., Failure Diagnosis Using Discrete-Event Models, *IEEE Transactions on Control Systems Technology*, 1996, vol. 4, no. 2, pp. 105–124.
3. Zaytoon, J. and Lafortune, S., Overview of Fault Diagnosis Methods for Discrete Event Systems, *Annual Reviews in Control*, 2013, vol. 37, no. 2, pp. 308–320.
4. Cassandras, Ch. and Lafortune, S., *Introduction to Discrete Event Systems*, 2nd ed., New York: Springer Science+Business Media, 2008.
5. Zhirabok, A.N., Kalinina, N.A., and Shumskii, A.E., Technique of Monitoring a Human Operator's Behavior in Man-Machine Systems, *Journal of Computer and Systems Sciences International*, 2018, vol. 57, no. 3, pp. 443–452.
6. Zhirabok, A.N., Kalinina, N.A., and Shumskii, A.E., Method for the Functional Diagnosis of Nondeterministic Finite State Machines, *Journal of Computer and Systems Sciences International*, 2020, vol. 59, no. 4, pp. 565–574.
7. Hartmanis, J. and Stearns, R., *The Algebraic Structure Theory of Sequential Machines*, New York: Prentice-Hall, 1966.
8. Zadeh, L.A., Fu, K.S., Tanaka, K., and Shimura, M., *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, New York: Academic Press, 1975.
9. Wee, W.G. and Fu, K.S., A Formulation of Fuzzy Automata and Its Applications as a Model of Learning Systems, *IEEE Trans. Syst. Science and Cybernetics*, 1969, vol. 5, no. 3, pp. 215–223.
10. Ulzutuev, I.E. and Maximov, A.A., The Properties of Subautomata Lattices of Fuzzy Semi-automata and Their Determinizators, *Vestnik of Saratov State Technical University*, 2015, vol. 79, no. 2, pp. 117–126. (In Russian.)
11. Voevodin, V.V. and Kuznetsov, Yu.A., *Matritsy i vychisleniya* (Matrices and Computations), Moscow: Nauka, 1984. (In Russian.)
12. Gruzlikov, A.M. and Kolesov, N.V., Discrete-Event Diagnostic Model for a Distributed Computational System. Independent Chains, *Autom. Remote Control*, 2016, vol. 77, no. 10, pp. 1805–1817.
13. Tuxi, T.M., Carvalho, L.K., Nunes, E.V.L., and Cunha, A.E., Diagnosability Verification Using LTL Model Checking, *Discrete Event Dynamic Systems*, 2022, vol. 32, no. 3, pp. 399–433.

This paper was recommended for publication by S.A. Krasnova, a member of the Editorial Board.

Received March 4, 2025, and revised April 17, 2025. Accepted April 29, 2025.

Author information

Shumsky, Aleksei Evgen'evich. Dr. Sci. (Eng.), Far Eastern Federal University, Vladivostok, Russia
✉ zhirabok@mail.ru
ORCID iD: https://orcid.org/0000-0002-5429-7482

Zhirabok, Aleksei Nilovich. Dr. Sci. (Eng.), Far Eastern Federal University, Vladivostok, Russia
✉ zhirabok@mail.ru
ORCID iD: https://orcid.org/0000-0001-5927-7117

Cite this paper

Shumsky, A.E., and Zhirabok, A.N., A Fault Diagnosis Method for Discrete-Event Systems Based on the Fuzzy Finite State Automaton Model. *Control Sciences* **2**, 31–41 (2025).

Original Russian Text © Shumsky, A.E., Zhirabok, A.N., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 38–49.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

CONSTRUCTING THE CES PRODUCTION FUNCTION BASED ON THE DISCRETE WEIBULL DISTRIBUTION

V. V. Kokov* and V. V. Sokolyanskiy**

Bauman Moscow State Technical University, Moscow, Russia

*✉ kokovvsevo@gmail.com, **✉ sokolyansky63@mail.ru

Abstract. This paper considers a probabilistic approach to obtaining the CES production function. It consists in calculating the mean and median of the Leontief function (the quantity of output) as a random variable depending on the capacities of production factors, i.e., the ratios of the factors to their per-unit values. The type of the cumulative distribution function of the minimum from a set of independent random variables is substantiated. Explicit expressions are derived for the mean and median of the quantity of output as CES functions when the factor capacities have (continuous) Weibull distributions. Discretely distributed production factors are considered using the example of a geometric law. An attempt is made to derive the CES function when the factor capacities have discrete Weibull distributions. The difficulties arising in the analytical use of the mean of the Leontief function are described.

Keywords: production function, CES production function, probabilistic approach, Weibull distribution, discrete Weibull distribution, geometric distribution, mean, median.

INTRODUCTION

Traditionally, production functions that establish the relationship between production factors $X_1,...,X_n$ and the quantity Q of output by an enterprise (or a country) are described in terms of the marginal rate of substitution S_{ij} and the elasticity of substitution σ_{ij} of a factor X_i by a factor X_j; by assumption, the factors take deterministic values [1]. In particular, for two factors X_1 and X_2, the property $\sigma_{12} = \text{const}$ is possessed by the CES (constant elasticity of substitution) function.

However, starting from the 1950s, a probabilistic approach to the description of production functions had stood out and was particularly developed in 1990–2015; for example, see [2–5]. The most significant achievement here was the development of a theoretical apparatus based on the following concepts: technology (idea), local production function, technology menu, and global production function [2]. We briefly explain the essence by the following example [3].

Let X_1 and X_2 be two production factors with some technological parameters x_1 and x_2, respectively. Consider the CES production function

$$Y = A\left(\psi\left(\frac{X_1}{x_1}\right)^{\theta} + (1-\psi)\left(\frac{X_2}{x_2}\right)^{\theta}\right)^{\frac{1}{\theta}},$$

where $A > 0$, $\psi \in (0, 1)$, and $\theta \in (-\infty, 0) \cup (0, 1)$ are constants.

A pair of parameter values (x_1, x_2) is called a *technology* or *idea*. The function Y with fixed values x_1 and x_2 is called the *local production function corresponding to the technology* (x_1, x_2).

Let a relation (*technology menu*) be imposed on the parameter values x_1 and x_2:

$$T_1(x_1)T_2(x_2) = N,$$

where $T_1(x_1)$ and $T_2(x_2)$ are some (unknown) functions of one variable and N is a constant.

In this example, the following problem arises naturally: given the values of the factors X_1 and X_2, find functions $T_1(x_1)$ and $T_2(x_2)$ such that the function Y will reach the largest values under the technology menu:

$$\begin{cases} Y = A\left(\psi\left(\frac{X_1}{x_1}\right)^{\theta} + (1-\psi)\left(\frac{X_2}{x_2}\right)^{\theta}\right)^{\frac{1}{\theta}} \to \max \\ T_1(x_1)T_2(x_2) = N. \end{cases}$$

(In this case, Y will be called the *global production function*.)

Using Lagrange's method of multipliers and the variable separation method, we can show that $F_1(x_1)=1-T_1(x_1)$ and $F_2(x_2)=1-T_2(x_2)$ are Weibull distribution functions. Researchers also studied the inverse problem [4]: reconstruct the global production function as a CES function from the parameters x_1 and x_2 distributed according to the Weibull law.

An alternative approach was later proposed by A.V. Mikheev [6] as follows. Denoting by x_i *the per-unit value of a factor* X_i (its quantity required to manufacture one product), he introduced the *capacity* Q_i of X_i as the ratio of the quantity of X_i to the per-unit value x_i:

$$Q_i=\frac{X_i}{x_i}.$$

With the factor capacities treated as random variables, the mean of the two-factor Leontief production function $Q=\min\{Q_1,Q_2\}$ was found through the double integral [6]:

$$EQ=\int_0^{+\infty} q_1\left(\int_{q_1}^{+\infty}(p_{Q_1Q_2}(q_1,q_2)+p_{Q_1Q_2}(q_2,q_1))dq_2\right)dq_1, \quad (1)$$

where $p_{Q_1Q_2}(q_1,q_2)$ stands for the joint density of the random variables Q_1 and Q_2. Based on formula (1), Mikheev established the following result: if Q_1 and Q_2 are independent and obey Weibull distributions with the same shape coefficient $\beta>0$, then EQ is expressed through the means EQ_1 and EQ_2 of Q_1 and Q_2 as the CES function

$$EQ=\left((EQ_1)^{-\beta}+(EQ_2)^{-\beta}\right)^{-\frac{1}{\beta}}.$$

Formula (1) leads to quite bulky calculations. However, another solution is possible: find the law (function or density) of distribution of the random variable $Q=\min\{Q_1,Q_2\}$ and derive the mean EQ by definition. If the random variables Q_1 and Q_2 were independent and identically distributed, this problem would turn into a well-known one of mathematical statistics: find the law of distribution of the minimum realization from a random sample with a given universe [7]. The advantage of this problem is the possibility to work with a random sample of any size *n* (i.e., *n* capacities $Q_1,\ldots,Q_n$ can be considered). Some modification of this problem is of interest for further research.

Note that only continuous models were considered in [2–6]; in reality, however, production factors or their capacities may be discrete variables. Here, we are concerned with the capacities of production factors as discretely distributed random variables and attempt to construct production functions on their basis. It is especially important to try reconstructing the CES function based on the discrete analog of the Weibull distribution using analytical methods.

According to the above considerations, we highlight several tasks:

- propose an effective method for finding the distribution law of the Leontief function from the capacities of production factors as independent random variables;
- show the possibility of obtaining the CES function from the means and medians of the capacities of *n* independent random production factors representing independent random variables with continuous Weibull distributions with the same shape coefficient;
- analyze discretely distributed capacities of production factors on the example of a geometric law;
- make an attempt to construct, by analytical methods, the CES function in the case of independent random capacities of production factors with discrete Weibull distributions with the same shape coefficient.

1. THE GENERAL PROPOSITION

Consider production with *n* non-fungible factors of capacities $Q_1,\ldots,Q_n$. For such factors, we can apply Leontief's production principle: the quantity of output is equal to the smallest of the capacities of the production factors used. In addition, we treat the capacities $Q_1,\ldots,Q_n$ as independent random variables.

The analysis below proceeds from the following result.

Proposition 1. *Let* $Q_1,\ldots,Q_n$ *be independent random variables, each having the distribution function*

$$F_{Q_i}(q)=\begin{cases} f_i(q), & q\ge b_i \\ 0, & q<b_i, \end{cases}$$

where b_i *are some numbers. Then the distribution function of the random variable*

$$Q=\min\{Q_1,\ldots,Q_n\}$$

is

$$F_Q(q)=1-\prod_{i=1}^{n}(1-F_{Q_i}(q)). \quad (2)$$

Proof.

$$F_Q(q)=P\{Q<q\}=1-P\{Q\ge q\}$$

$$=1-P\{\min\{Q_1,\ldots,Q_n\}\ge q\}=1-P\{Q_1\ge q,\ldots,Q_n\ge q\}.$$

But $P\{Q_1 \ge q,...,Q_n \ge q\} = P\{Q_1 \ge q\} \cdot ... \cdot P\{Q_n \ge q\}$ since $Q_1,...,Q_n$ are independent. Thus,

$$F_Q(q) = 1 - P\{Q_1 \ge q\} \cdot ... \cdot P\{Q_n \ge q\},$$

which finally gives formula (2). ♦

Remark. In this paper, we also deal with discretely distributed random variables $Q_1,\ldots,Q_n$ with integer values 1, 2, 3, ... or 0, 1, 2, ... For such random variables, Proposition 1 remains valid. (For the sake of simplicity, imagine that b_i are integers and integers are selected from the set $q \ge b_i$.)

2. THE CASE OF CONTINUOUS VARIABLES

Let us represent Q_1 and Q_2 as the capacities of some production factors. Suppose that they are two independent continuous random variables obeying the Weibull laws [8] with the same shape coefficient $\beta > 0$ and coefficients $\alpha_1 > 0$ and $\alpha_2 > 0$, respectively:

$$F_{Q_1}(q) = \begin{cases} 1 - e^{-\alpha_1 q^\beta}, & q \ge 0 \\ 0, & q < 0, \end{cases}$$

$$F_{Q_2}(q) = \begin{cases} 1 - e^{-\alpha_2 q^\beta}, & q \ge 0 \\ 0, & q < 0. \end{cases}$$

Assume that Q (the quantity of output) depends on the capacities Q_1 and Q_2 by Leontief's production principle:

$$Q = \min\{Q_1, Q_2\}.$$

Applying Proposition 1 yields

$$F_Q(q) = \begin{cases} 1 - e^{-\alpha q^\beta}, & q \ge 0 \\ 0, & q < 0, \end{cases}$$

where $\alpha = \alpha_1 + \alpha_2$. Thus, the random variable Q has a Weibull distribution. Its mean is given by [8]

$$EQ = \alpha^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right) = \left(\alpha_1 + \alpha_2\right)^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right), \quad (3)$$

where Γ denotes the gamma function.

We calculate the coefficients α_1 and α_2 from

$$EQ_1 = \alpha_1^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right),$$

$$EQ_2 = \alpha_2^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right)$$

(the formulas for the means EQ_1 and EQ_2 of the random variables Q_1 and Q_2) and substitute the results into (3), thereby obtaining

$$EQ = \left((EQ_1)^{-\beta} + (EQ_2)^{-\beta}\right)^{-\frac{1}{\beta}}. \quad (4)$$

This formula defines the CES production function.

In the following, the special case $\beta = 1$ is also of interest. In this case, the Weibull distribution (for Q_1, Q_2, and Q) turns into the exponential one, and the expression (4) takes the form

$$EQ = \frac{EQ_1 EQ_2}{EQ_1 + EQ_2}. \quad (5)$$

These calculations can be generalized to the case of n factors. In addition, other characteristics of random variables (e.g., medians) can be considered instead of means.

Proposition 2. *Let Q_i $(i = 1,...,n)$ be the capacities of production factors represented as independent continuous random variables with Weibull distributions with the same shape coefficient $\beta > 0$ and coefficients $\alpha_1 > 0,\ldots,\alpha_n > 0$:*

$$F_{Q_i}(q) = \begin{cases} 1 - e^{-\alpha_i q^\beta}, & q \ge 0 \\ 0, & q < 0. \end{cases}$$

In addition, let the quantity of output Q be determined by Leontief's production principle:

$$Q = \min\{Q_1,...,Q_n\}.$$

Then the mean EQ and median M of Q are expressed through the means EQ_i and medians M_i of Q_i, respectively, as CES production functions:

$$EQ = \left((EQ_1)^{-\beta} + ... + (EQ_n)^{-\beta}\right)^{-\frac{1}{\beta}}, \quad (6)$$

$$M = \left(M_1^{-\beta} + ... + M_n^{-\beta}\right)^{-\frac{1}{\beta}}. \quad (7)$$

P r o o f. Utilizing Proposition 1, we obtain the Weibull distribution function

$$F_Q(q) = \begin{cases} 1 - e^{-\alpha q^\beta}, & q \ge 0 \\ 0, & q < 0, \end{cases}$$

where $\alpha = \alpha_1 + \ldots + \alpha_n$. The mean EQ and median M of the random variable Q are given by

$$EQ = \alpha^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right) = \left(\alpha_1 + ... + \alpha_n\right)^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right),$$

$$M = \alpha^{-\frac{1}{\beta}} (\ln 2)^{\frac{1}{\beta}} = \left(\alpha_1 + ... + \alpha_n\right)^{-\frac{1}{\beta}} (\ln 2)^{\frac{1}{\beta}}.$$

By analogy, the means EQ_i and medians M_i of the random variables Q_i are given by

$$EQ_i = \alpha_i^{-\frac{1}{\beta}} \Gamma\left(\frac{1}{\beta} + 1\right),$$

$$M_i = \alpha_i^{-\frac{1}{\beta}} (\ln 2)^{\frac{1}{\beta}}.$$

Calculating the coefficients α_i from these formulas and substituting the results into the expressions for EQ and M, we finally arrive at (6) and (7). ♦

3. THE CASE OF DISCRETE VARIABLES

Now we attempt to represent the capacities of production factors as random variables with discrete distributions.

3.1. An Example: Geometric Distribution

Let Q_1 and Q_2 be the capacities of production factors described by two independent random variables with geometric distributions with parameters p_1 and p_2, respectively (the probabilities of success in single trials). Here, we understand the distribution in the following sense: the random variable is the trial number with the first success (possible values are $j = 1,2,...$).

Then the probabilities of failure in single trials are $q_1 = 1 - p_1$ and $q_2 = 1 - p_2$, respectively.

The distribution functions of the random variables Q_1 and Q_2 have the form

$$F_{Q_1}(j) = P\{Q_1 < j\} = 1 - q_1^{j-1},$$

$$F_{Q_2}(j) = P\{Q_2 < j\} = 1 - q_2^{j-1}.$$

Let the quantity of output Q be determined by Leontief's production principle:

$$Q = \min\{Q_1, Q_2\}.$$

Proposition 1 yields

$$F_Q(j) = 1 - q_0^{j-1},$$

where $q_0 = q_1 q_2$. Obviously, the random variable Q has a geometric distribution with the parameter $p_0 = 1 - q_0$. We express p_0 through the parameters p_1 and p_2:

$$p_0 = 1 - q_1 q_2 = 1 - (1 - p_1)(1 - p_2) = p_1 + p_2 - p_1 p_2.$$

As is well known,

$$EQ_i = \frac{1}{p_i},\ i = 1,2, \tag{8}$$

$$EQ = \frac{1}{p_0} = \frac{1}{p_1 + p_2 - p_1 p_2}. \tag{9}$$

Calculating p_1 and p_2 from (8) and substituting the results into formula (9), we obtain

$$EQ = \frac{EQ_1 EQ_2}{EQ_1 + EQ_2 - 1}. \tag{10}$$

Note that, with another definition of the geometric distribution used, the random variable is the number of failures before the first success. In this case,

$$EQ = \frac{EQ_1 EQ_2}{EQ_1 + EQ_2 + 1}. \tag{11}$$

Here is an elementary example of describing the capacities Q_1 and Q_2 of production factors by geometrically distributed random variables. Suppose that a trial to manufacture a single indivisible product requires consuming the unit capacity of either factor 1 or factor 2. A single trial may be successful (the product passes inspection and testing) or not (otherwise). Let p_1 and p_2 denote the probabilities of success when using factors 1 and 2, respectively. By assumption, the trial number does not affect the probability of success.

The number of the first successful trial to manufacture a single product using the selected factor is the realization of its capacity as a random variable, and this variable obeys the geometric distribution by definition. Obviously, it is advantageous to use the factor with the minimum number of the first success (a reference to Leontief's principle).

Of interest is some similarity of formulas (10), (11) with formula (5) obtained for exponentially distributed factors.

As is well known, the geometric distribution is a discrete analog of the exponential distribution. Let some random variable Q have an exponential distribution with the density function

$$p_Q(q) = \begin{cases} \lambda e^{-\lambda q}, & q \geq 0 \\ 0, & q < 0, \end{cases}$$
$$\lambda > 0.$$

Consider the random variable $Y = \lceil Q \rceil$, i.e., the ceiling of the variable Q. For natural numbers $j = 1,2,...$, we have

$$P\{Y = j\} = P\{j - 1 < Q \leq j\} = \int_{j-1}^{j} p_Q(q)dq,$$

implying

$$P\{Y = j\} = (1 - e^{-\lambda})e^{-\lambda(j-1)}.$$

With denoting $q_0 = e^{-\lambda} = 1 - p_0$, it follows that

$$P\{Y = j\} = (1 - q_0)q_0^{j-1} = p_0(1 - p_0)^{j-1}.$$

Thus, the variable Y has a geometric distribution with the parameter $p_0 = 1 - e^{-\lambda}$, being interpreted as the number of the first successful trial.

Note that the floor of $\lfloor Q \rfloor$ also obeys a geometric law with the parameter $p_0 = 1 - e^{-\lambda}$, meaning the number of failures before the first success.

3.2. Discretization of the Weibull Distribution and an Attempt to Construct the CES Function

Let us discretize the Weibull distribution by analogy. In this case, the density of the random variable Q has the form

$$p_Q(q) = \begin{cases} \alpha\beta q^{\beta-1} e^{-\alpha q^{\beta}}, & q \geq 0 \\ 0, & q < 0, \end{cases}$$
$$\beta > 0,\ \alpha > 0.$$

Consider the random variable $Y = \lfloor Q \rfloor$. For $j = 0, 1, 2, \ldots$, we obtain

$$P\{Y = j\} = P\{j \le Q < j+1\} = \int_{j}^{j+1} p_Q(q)dq$$

and, after straightforward transformations,

$$P\{Y = j\} = e^{-\alpha j^{\beta}} - e^{-\alpha(j+1)^{\beta}}. \quad (12)$$

Then the distribution function of the random variable Y is given by

$$F_Y(j) = P\{Y < j\} = \sum_{k=0}^{j-1} P\{Y = k\} = 1 - e^{-\alpha j^{\beta}}, \quad (13)$$

representing the desired discrete Weibull distribution of type 1 [9, 10].

Let us calculate the median M of the random variable Y. For this purpose, we introduce the quantile Q_γ of level γ; its unrounded value is the solution of the equation

$$F_Y(Q_\gamma) = \gamma.$$

In view of formula (13), this equation becomes

$$1 - e^{-\alpha Q_\gamma^{\beta}} = \gamma.$$

After trivial transformations we obtain

$$Q_\gamma = \left(-\frac{\ln(1-\gamma)}{\alpha}\right)^{\frac{1}{\beta}},$$

and the unrounded value of the median is

$$M = Q_{1/2} = \left(\frac{\alpha}{\ln 2}\right)^{-\frac{1}{\beta}}. \quad (14)$$

Consider now the mean of the random variable Y given formula (12):

$$EY = \sum_{j=0}^{\infty} jP\{Y = j\} = \sum_{j=0}^{\infty} j\left(e^{-\alpha j^{\beta}} - e^{-\alpha(j+1)^{\beta}}\right). \quad (15)$$

This series is often calculated numerically [9]; in this paper, we endeavor to derive an analytical expression.

Assuming the convergence of the series (15), we open brackets in the expansion and combine the neighboring similar terms to get

$$EY = \sum_{j=1}^{\infty} (e^{-\alpha})^{j^{\beta}}. \quad (16)$$

Here we study the case $\beta > 1$. Then each term of the series (16) is smaller than the corresponding term of the convergent geometric progression series $\sum_{j=1}^{\infty} e^{-\alpha j}$; therefore, the series (16) will converge as well.

Of theoretical interest is the case $\beta = 2$ (a discrete analog of the Rayleigh distribution). In the remainder of the paper, we focus on this case, denoting the series (16) by $u(\alpha)$:

$$EY = u(\alpha) = \sum_{j=1}^{\infty} e^{-\alpha j^2} = (e^{-\alpha})^{1^2} + (e^{-\alpha})^{2^2} + (e^{-\alpha})^{3^2} + \ldots \quad (17)$$

For the analysis, it seems reasonable to introduce the theta function

$$\theta(s) = \sum_{j=-\infty}^{\infty} (e^{-\pi s})^{j^2}$$

and the function

$$w(s) = \sum_{j=1}^{\infty} (e^{-\pi s})^{j^2} = \frac{1}{2}(\theta(s) - 1).$$

According to [11], the following functional equation is valid:

$$w(s) = \frac{1}{\sqrt{s}} w\left(\frac{1}{s}\right) + \frac{1}{2}\left(\frac{1}{\sqrt{s}} - 1\right).$$

Letting $\pi s = \alpha$, we relate the functions $u(\alpha)$ and $w(s)$ by

$$u(\alpha) = w(s) = w\left(\frac{\alpha}{\pi}\right);$$

hence,

$$u(\alpha) = \sqrt{\frac{\pi}{\alpha}} \cdot u\left(\frac{\pi^2}{\alpha}\right) + \frac{1}{2}\left(\sqrt{\frac{\pi}{\alpha}} - 1\right). \quad (18)$$

For the scale coefficients $0 < \alpha \lesssim 2$, the first term on the right-hand side of (18) can be neglected. In this case, we have the approximate equality

$$EY \approx \frac{1}{2}\left(\sqrt{\frac{\pi}{\alpha}} - 1\right), \quad 0 < \alpha \lesssim 2,$$

which can be written as

$$EY + \frac{1}{2} \approx \frac{\sqrt{\pi}}{2}\alpha^{-\frac{1}{2}}, \quad 0 < \alpha \lesssim 2. \quad (19)$$

For $\alpha \gtrsim 2$, the analysis of the expansion (17) can be restricted to the first term and, consequently,

$$EY \approx e^{-\alpha}, \quad \alpha \gtrsim 2.$$

Based on Proposition 1 and the above considerations for the discrete Weibull distribution, we formulate the following result.

Proposition 3. *Let* Q_i $(i=1,...,n)$ *be the capacities of production factors represented as independent random variables with discrete Weibull distributions with the same shape coefficient* $\beta>0$ *and coefficients* $\alpha_1>0,\ldots,\alpha_n>0$:

$$F_{Q_i}(j)=1-e^{-\alpha_i j^{\beta}},\quad j=0,\,1,\,2,\,...$$

In addition, let the quantity of output Q *be determined by Leontief's production principle:*

$$Q=\min\{Q_1,...,Q_n\}.$$

Then the unrounded median M (14) *of the variable* Q *is related to the unrounded medians* M_i *of the variables* Q_i *through a CES function:*

$$M=\left(M_1^{-\beta}+...+M_n^{-\beta}\right)^{-\frac{1}{\beta}}.$$

Moreover, if $\beta=2$ *and* $0<\alpha\lesssim 2$, *where* $\alpha=\alpha_1+...+\alpha_n$, *then the mean* EQ *of* Q *can be approximately related to the means* EQ_i *of* Q_i *through a CES function:*

$$EQ+\frac{1}{2}\approx\left((EQ_1+\frac{1}{2})^{-2}+...+(EQ_n+\frac{1}{2})^{-2}\right)^{-\frac{1}{2}}.$$

Note to Proposition 3. Under the constraint $0<\alpha=\alpha_1+\ldots+\alpha_n\lesssim 2$, the inequalities $0<\alpha_i\lesssim 2$, $i=1,...,n$, hold immediately. Hence, the means EQ_i can be expressed in the desired approximate form (19).

CONCLUSIONS

In this paper, we have obtained CES functions for the means and medians of the capacities of *n* production factors in the case where the capacities are represented as independent random variables with continuous Weibull distributions with the same shape coefficient.

We have proposed to consider discretely distributed capacities of production factors on the example of a geometric law. In this case, according to Leontief's production principle, it is advantageous to use the factor with the minimum number of the first successful trial (product manufacturing).

Also, we have endeavored to construct the CES function in the case of independent random factor capacities with discrete Weibull distributions with the same shape coefficient. As a result, the unrounded values (14) of the medians of factor capacities and the median of the quantity of output have been successfully related. However, difficulties arise when establishing a relationship between the means of these variables.

The main challenge in this study has been to derive, in analytical terms, the mean of a random variable distributed according to the discrete Weibull law. In the special case $\beta=2$ (the shape coefficient), it is possible to introduce the theta function and compile a functional equation. With some restrictions on the values of the scale coefficient of the distribution, it is possible to neglect some part of the functional equation, thereby approximating the required mean (see formula (19)) and deriving the CES function.

In the *general case* (no restrictions on the distribution coefficients), a still open issue is the possibility of relating the means of the capacities of production factors represented as random variables with discrete Weibull distributions with the same shape coefficient.

REFERENCES

1. Gorbunov, V.K., *Proizvodstvennye funktsii: teoriya i postroenie* (Production Functions: Theory and Construction), Ulyanovsk: Ulyanovsk State University, 2013. (In Russian.)
2. Jones, C.I., The Shape of Aggregate Production Functions and the Direction of Technical Change, *Quarterly Journal of Economics*, 2005, vol. 120, no. 2, pp. 517–549.
3. Growiec, J., Production Functions and Distributions of Unit Factor Productivities: Uncovering the Link, *Economics Letters*, 2008, vol. 101, no. 1, pp. 87–90.
4. Growiec, J., A Microfoundation for Normalized CES Production Functions with Factor-augmenting Technical Change, *Journal of Economic Dynamics and Control*, 2013, vol. 37, no. 11, pp. 2336–2350.
5. Matveenko, V.D., "Anatomy" of the Production Function: Technological Menu and Selection of the Best Technology, *Economics and Mathematical Methods*, 2009, vol. 45, no. 2, pp. 85–95. (In Russian.)
6. Mikheev, A.V., Probabilistic Approach to Determining Production Functions, *Vestnik of Astrakhan State Technical University. Series: Management, Computer Science and Informatics*, 2021, no. 4, pp. 82–94. (In Russian.)
7. Goryainov, V.B., Pavlov, I.V., Tsvetkova, G.M., et al., *Matematicheskaya statistika* (Mathematical Statistics), Zarubin, V.S., and Krishchenko, A.P., Eds., 3rd ed., Moscow: Bauman Moscow State Technical University, 2008. (In Russian.)
8. Pechinkin, V.A., Teskin, O.I., Tsvetkova, G.M., et al., *Teoriya veroyatnostei* (Probability Theory), Zarubin, V.S., and Krishchenko, A.P., Eds., Moscow: Bauman Moscow State Technical University, 1998. (In Russian.)
9. Rinne, H., *The Weibull Distribution: A Handbook*, New York: Chapman and Hall/CRC, 2008.
10. Barbiero, A., Discrete Weibull Distributions (Type 1 and 3), *The Comprehensive R Archive Network*, 2025. URL: https://cran.r-project.org/web/packages/DiscreteWeibull/DiscreteWeibull.pdf.
11. Woit, P., *Fourier Analysis Notes*, New York: Department of Mathematics, Columbia University, 2020. URL: https://www.math.columbia.edu/~woit/fourier-analysis/fouriernotes.pdf.

This paper was recommended for publication by M.I. Geraskin, a member of the Editorial Board.

Received February 21, 2025,
and revised April 27, 2025.
Accepted April 29, 2025.

Author information

Kokov, Vsevolod Vyacheslavovich. Student, Bauman Moscow State Technical University, Moscow, Russia
✉ kokovvsevo@gmail.com
ORCID iD: https://orcid.org/0009-0008-4331-5148

Sokolyanskiy, Vasily Vasil'evich. Associate Professor, Bauman Moscow State Technical University, Moscow Samara, Russia
✉ sokolyansky63@mail.ru
ORCID iD: https://orcid.org/0000-0002-6636-4638

Cite this paper
Kokov, V.V. and Sokolyanskiy, V.V., Constructing the CES Production Function Based on the Discrete Weibull Distribution. *Control Sciences* **2,** 42–48 (2025).

Original Russian Text © Kokov, V.V., Sokolyanskiy, V.V., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 50–57.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com

A PROCEDURE FOR ASSESSING SECURITY UPDATES IN INDUSTRIAL SYSTEMS

K. V. Semenkov* and V. G. Promyslov**

*,**Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

*✉ semenkov@ipu.ru, **✉ vp@ipu.ru

Abstract. This paper is devoted to the problem of applying cybersecurity updates (patches) for the software of instrumentation and control systems (ICS) with a long lifecycle. The problem is considered for the system operation stage. The main focus is on the large number of vulnerabilities found in software, the complexity of analyzing the impact of a vulnerability on system security, and the requirements for testing the compatibility of updates and software certification after changes have been made. Based on the Failure Mode and Effects Analysis (FMEA), a procedure is proposed to simplify the analysis of the impact of a vulnerability on cybersecurity. This procedure considers a smaller set of attack scenarios rather than each vulnerability separately. The analysis of attack scenarios also covers the effect of security measures. The procedure includes simple criteria for applying security updates based on the analysis results. An example of vulnerability analysis using this procedure is provided.

Keywords: vulnerability, patch, risk assessment, instrumentation and control system (ICS), cybersecurity, criterion.

INTRODUCTION

A main way of conducting cyberattacks is to exploit vulnerabilities in software. Despite advances in software development and testing technologies, the complexity of programs makes it impossible to guarantee that software is free of vulnerabilities. Intruders focus their resources on finding and exploiting vulnerabilities, whereas software developers and users are interested in finding and promptly fixing these vulnerabilities, and releasing and installing appropriate software security updates (patches). In general, a patch is often understood as a very wide range of software changes [1–4], which are either characterized by the approach to this patch as a/an process/object or related to the scope or nature of software changes. In informal communication of IT experts, one can also meet other similar terms, e.g., update, bugfix, or hotfix. For the purposes of this paper, let us assume the following.

Definition. A security update (patch) is a modification to installed software intended to eliminate software vulnerabilities without changing other functional characteristics of the software. ♦

A great deal of work has been done globally to accumulate information about vulnerabilities, and there are publicly available and constantly updated databases: CVE (USA) [5], the Data Bank of Information Security Threats (Russia, managed by the Federal Service for Technical and Export Control (FSTEC)) [6], CERT-FR (France) [7], and others. For each vulnerability on the list, the databases necessarily contain its description, impact assessment, and recommendations to eliminate the vulnerability or mitigate its negative impact. Software developers release security updates, which can often be installed automatically.

The experience gained in the world is systematized in international and national standards and methodological documents on the application of patches; for example, see [3, 8, 9]. The guidelines and recommendations of these documents are generally reduced to the following steps:

1) permanently monitor vulnerabilities in the software used;

2) analyze newly discovered vulnerabilities and assess cybersecurity risks;

3) determine further actions depending on the results of the risk assessment: accept the risk, eliminate the risk (apply a patch), etc.;

4) under a positive decision to install an update:
   a) develop a plan for applying the update;

b) check the integrity and confidence of the update;
c) test the update;
d) install the update;
e) check the software status and configuration after the installation.

As we believe, however, the issues of practical application of the available data, primarily related to the large volume of analyzed information and its reliability, have not yet been fully settled.

In this paper, the problem of applying security updates [10, 11] in systems with a long lifecycle will be considered on the example of an instrumentation and control system (ICS), and a new solution method will be proposed. Below, a system with a long lifecycle is understood as a system whose operation and support stage lasts for several years or even decades.

According to the guidelines of regulatory and methodological documents [10–13], vulnerability identification, analysis, and assessment should be conducted throughout the entire lifecycle of a protected system. Since cybersecurity resources and the context of viewing the system differ significantly between lifecycle stages, vulnerability elimination problems and methods for addressing them also differ. This paper focuses on the issues of vulnerability assessment during the exploitation stage under the following assumption: at the end of the development stage, the developer has closed known vulnerabilities and applied adequate protection measures to mitigate the risk to an acceptable level. For the development stage, there is a diverse set of recommendations for secure software development [14].

Much attention is paid to the operation stage because, as our experience shows, the problem of vulnerability management most fully manifests itself at the operation stage and becomes more complicated over time. This is primarily due to the integral effect of several factors: the accumulation of detected vulnerabilities in the components used, the end of the developer's support period for some components, and the obsolescence of information security technologies embedded in the system design.

As the object of study, we choose complex software systems, i.e., sets of programs [15] with special system components and third-party components of general application. Assume that the total number of components in the system is sufficiently large: for simple systems, the patching problem seems to be not very serious due to a moderate number of vulnerabilities, which can be promptly monitored and eliminated in the operation process. As shown by the practice, "simple" systems for ICSs consist of at most a single computer; then the number of assets and their links allows describing the emerging security relationships by an access control model, which can be used in risk assessment for assets associated with detectable vulnerabilities.

Risk management in industrial facilities and the installation of updates for digital safety systems are important, both scientifically and practically. The problem of risk assessment for the ICS of nuclear power plants (NPPs) was reviewed in [16]; a comprehensive survey of the recent (2002–2020) publications on patch management was given in [17]. The contribution of this paper is the detailed description of top-level techniques (such as [1, 10]) and a novel, industrial control system-oriented, set of actions for deciding on the installation of updates.

The problem of patch management will be considered in terms of the functions performed by the system rather than the vulnerability of a particular component for which a patch is available. For example, the main function of an ICS is to control an industrial facility. Then the purpose of installing a patch for the operating system (OS) of a computer within the control system is not to protect the OS but to mitigate the risk of the facility's uncontrollability in case of vulnerability exploitation. For the solution, based on *Failure Mode and Effects Analysis* (FMEA) [18], we propose a risk-oriented method with criteria for applying updates. In addition to managing vulnerabilities by their types, the idea is to consider explicitly the impact of protection measures on the realizability of attacks by an intruder with certain capabilities. The approach described below allows comparing the newly discovered vulnerabilities with known ones from some classifier (e.g., the *Common Weakness Enumeration* (CWE) [19]) and answering the following question: Does the system have "an immunity" (a barrier) against a new vulnerability? Hereinafter we will understand a barrier as a certain set of protection measures that guarantee security in a definite attack scenario.

1. THE PECULIARITIES OF USING THE THREAT MODEL IN PATCH INSTALLATION

Most risk-oriented approaches to patch management involve risk assessment techniques formulated in the ISO/IEC 27005 standard [20]. According to these techniques, the analysis of a threat model, including vulnerabilities, threats, and an intruder, mainly influences the decision of risk acceptability or unacceptability and, consequently, the decision to patch the system. There are many methods for describing threat model elements and compiling their taxonomy; below we will discuss the most appropriate ones for risk as-

sessment in complex industrial systems during the operation stage. Let us begin with the individual components of the threat model.

1.1. Vulnerability Analysis

Following the definition of a vulnerability from the ISO/IEC 27000 standard and FSTEC methodological documents [12], vulnerability is "weakness of an asset or control that can be exploited by one or more threats."

Vulnerabilities may have different nature. They can be related to the system properties embedded during development (weaknesses in the defense-in-depth architecture or cross-domain communication, implementation errors) or can appear due to incorrect application of protection measures (e.g., passwords). Vulnerability analysis is intended to establish the extent to which vulnerabilities can affect the security of the system and the assessment of confidence in the protection measures implemented [21]. Patches must be applied to a system if the vulnerability analysis reveals an unacceptable level of information security risk to the system (see the guidelines [9], *Fig. 3.1*). Let us demonstrate the problems arising in vulnerability analysis. For this purpose, consider the use of methodological guidelines for analyzing and applying patches in more detail.

The first problem that needs to be highlighted is the scale of the system. As mentioned above, a complex software system includes a large number of heterogeneous components and third-party applications, and cybersecurity requires monitoring a large number of vulnerabilities associated with both special system components and third-party products (e.g., vulnerabilities in the operating system, database management systems, web servers, interpreters, etc).

The number of newly discovered vulnerabilities increases every year. For example, Table 1 presents the corresponding figures for CVE and the FSTEC Data Bank in 2021–2023.

For a complex system, the flow of vulnerabilities can amount to tens or hundreds of vulnerabilities per day, and even the initial analysis of new vulnerabilities can require significant resources and costs for an organization.

Table 1

The number of vulnerabilities added to CVE and FSTEC Data Bank annually, in thousand

Database	2021	2022	2023
FSTEC Data Bank	6.4	7.5	9.1
CVE [22]	20.2	25.0	29.0

The next problem to be emphasized is that in all databases, vulnerabilities are described in a relatively free form, without a generally accepted standard. Descriptions can be either very brief or overly detailed, making their analysis even more complicated. Here are some examples of unsuccessful descriptions (Table 2). The CVE-2018-19932 vulnerability is described with many technical details (may be of interest only to software developers); the CVE-2023-36762 vulnerability has a too general characterization; finally, the CVE-2021-30618 vulnerability is included in the database almost without essential information.

There are works aimed at automating vulnerability description analysis (e.g., see [23, 24]), including those with machine learning algorithms. However, to the best of our knowledge, no available tools completely automate the analysis of real systems in practice.

The description problem is aggravated by the language barrier, which has a complex character. First, most vulnerability descriptions in international databases are written in English, fluently managed by far from all authors of such descriptions. In other words, even at the initial description phase, the essence of a vulnerability may be distorted or incompletely stated. Second, a large amount of information about vulnerabilities is transferred from international open databases to national ones, where the descriptions are usually moderated and translated into the local language. Thus, after such manipulations, national vulnerability databases may contain additional errors and inaccuracies that complicate vulnerability analysis and, vice versa, expanded and clarified descriptions. However, in the latter case, the developer of a component with an open vulnerability often loses feedback from the moderators: all clarifications made are available only in the national language.

Table 2

Examples of unsuccessful vulnerability descriptions

Vulnerability	Description
CVE-2018-19932	An issue was discovered in the Binary File Descriptor (BFD) library (aka libbfd), as distributed in GNU Binutils through 2.31. There is an integer overflow and infinite loop caused by the IS_CONTAINED_BY_LMA macro in elf.c.
CVE-2023-36762	Microsoft Word remote code execution vulnerability.
CVE-2021-30618	Inappropriate implementation in DevTools.

Therefore, an organization needs a sufficiently large staff of experts to monitor and analyze such a volume of vulnerabilities independently; experts will perform a professional analysis of this poorly structured information and assess the risk associated with the vulnerability to the protected system.

A possible way to reduce the amount of information analyzed is to prioritize vulnerabilities and focus on the most critical ones. The *Common Vulnerability Scoring System* (CVSS) [25–27] is one of the most well-known and widespread criticality assessment scales. This scoring system includes three groups of metrics: basic, temporal, and contextual. According to our experience, the last two either do not contain information or the information is specific for a particular scenario of program application. Therefore, we will use only the basic metrics. For CVSS 3.0 [26], they include:

- the attack vector (e.g., a network attack, a local attack, physical access, an attack on a related network protocol);
- the complexity of the attack;
- the necessary access rights to exploit the vulnerability;
- participation of a "normal" user to exploit the vulnerability;
- the possibility that the attack's consequences will go beyond the system under study;
- the impact on the availability, confidentiality, and integrity of information resources controlled by the program in which the vulnerability is detected.

The basic metrics mainly reflect the properties of a software component, and the factors of the operating environment are considered only conditionally. Their values, calculated via expertise, are given in vulnerability databases.

Therefore, the basic metrics are primarily intended for developers and users of a separate software component and ignore its role in the information system. To perform a complete CVSS vulnerability analysis for further risk assessment, one should calculate the remaining groups of metrics or use other metrics and vulnerability databases that reflect the above aspects [2, 28]. Also, it may be necessary to recalculate the values of basic metrics to consider the specifics of a particular organization or system under study.

The transition from individual vulnerabilities to their classes seems to reduce the labor costs of risk analysis and assessment. Vulnerabilities can be classified in different ways depending on the subject matter and the level of detail of the system. In general, there are the following types of vulnerabilities reflecting the nature of assets [13]:

– hardware vulnerabilities,
– software vulnerabilities,
– network vulnerabilities.

This paper deals only with software vulnerabilities because security updates are mainly focused on them.

To pass from the separate implementations of vulnerabilities (unique in most cases) to their types, we apply the approach [14], linking vulnerabilities to program weaknesses. Let us take CWE [19], one of the most famous classifiers of weaknesses.

This list is a hierarchical, freely augmentable taxonomy of software and hardware flaws that can be used in security analysis tools.

The classifier is a multilevel tree with four levels of weaknesses: Root level, Base level, Class level, and Variant level. To facilitate user work, CWE contains the so-called views, i.e., a set of CWE records intended for specific tasks (e.g., software development, hardware development, and research).

Vulnerabilities in CWE are classified manually by experts; according to practice, the opinions of experts differ in many cases [29]. Also, research into fully automatic classification is ongoing, but without unambiguous results available so far (e.g., see [23]).

The CWE catalog can be used to create a secure development system. As we believe, however, it is of little utility for vulnerability classification to create a protection system due to an implicit relationship between the CWE class, attack methods (e.g., attacks from the CAPEC classifier [30]), and the intruder model. Indeed, the same weakness can be used in different attack scenarios by different intruders and cause different consequences.

1.2. Analyzing Threats and Characteristics of Intruders

In the context of cybersecurity, a threat can be defined as a set of conditions and factors that create a potential or real danger of violating information security [12].

Threat types are usually correlated to the way of exploiting vulnerabilities, each with different implications and prerequisites; as a rule, threat types can be associated with a violation of cybersecurity properties in one of the reference models (e.g., the Confidentiality–Integrity–Availability (CIA) model).

The mapping of cybersecurity properties to system properties is individual for a particular system; depending on the system under analysis, each threat type may affect any system property (reliability, availability, maintainability, and safety). There are several common classifications of threat types:

– The Data Bank of Information Security Threats (FSTEC, Russia) [6],
– MITRE ATT&CK [31],
– Microsoft's STRIDE [32].

An intruder (threat agent, attacker) is an active element (subject) in a system that attempts to exploit a vulnerability. Examples are hackers, computer criminals, terrorists, industrial spies, and insiders [33]. A detailed classification of intruders is provided in the catalogs of FSTEC, CAPEC, MITRE ATT&CK, etc. [12, 30, 31]. Each classifier contains a set of attributes for intruders, e.g., type, the levels of competence and equipment, and the purpose of attack. Within the FSTEC model, intruders are further divided into external and internal. ISO/IEC 27000 and MITRE ATT&CK attribute an intruder by the purpose of attack: obtaining money, undermining reputation, gaining a competitive advantage, etc.

As an example of attribution according to FSTEC documents, we present the types of intruders and their competence levels.

The types of intruders according to FSTEC are:

- special services of foreign countries;
- terrorist and extremist groups;
- criminal groups (criminal structures);
- natural persons (hackers);
- competing organizations;
- developers of software and programmable digital items;
- suppliers of software and programmable digital items for supporting systems;
- providers of communication services and computing services;
- persons engaged for installation, adjustment, testing, commissioning, and other types of work;
- persons ensuring the operation of systems and networks or supporting systems of the operator (administration, security guards, cleaners, etc.);
- authorized users of systems and networks.

The competence levels (H1–H4) of an intruder according to the Russian regulator FSTEC are:

- basic capabilities for realizing information security threats (H1);
- increased capabilities for realizing information security threats (H2);
- medium capabilities for realizing information security threats (H3);
- high capabilities for realizing information security threats (H4).

This list may be supplemented by other intruder types, considering the peculiarities of the field where systems operate and the connection between the system under analysis and its environment.

1.3. Analysis of Threat Model Components: Some Conclusions

According to the aforesaid, clearly, the work on analyzing threat model components and assessing the risk from vulnerability exploitation by intruders requires the regular participation of experts with rich knowledge and skills in programming and information security risk assessment, both at the system level and at the level of separate components.

This work is very time-consuming and goes beyond the functions related to risk analysis for vulnerabilities. If a vulnerability needs to be eliminated by applying a patch, the system owner faces additional work and problems.

2. PATCH MANAGEMENT PROBLEMS

In Section 1, we have described the basic steps for deciding whether to patch or not, as well as the related problems. However, the difficulties do not stop there: having decided to patch, the system owner deals with new problems due to the complexity of this class of systems:

- The security update of a software component within a system is often released not separately but as part of a new version of the component. In this case, the functionality of the new version may require the additional testing of the component within the system. Replacing the existing version of a software component may terminate some ICS functions, causing the need to modify ICS software.
- Software components within a software system are interconnected by a chain of dependencies. Dependencies can be both horizontal (e.g., at the level of application software components) and vertical (at the level of OS components). Replacing any key component may entail replacing the rest and, in the worst case, replacing all components in the dependency chain. For a system with a long lifecycle, some components in a dependency chain may be no longer supported by the developer (no new versions exist for them), and the update in this case cannot be performed by simply passing to a new version.
- ICSs are characterized by a long lifecycle (the operation period may reach decades) and strict requirements for the procedure of their development and testing. Given the high rate of discovering new vulnerabilities, it seems natural to assume that new vulnerabilities will be found in the ICS software environment during the time between the release of the ICS software version and the launch of the system after commissioning. Also note that the suppliers of third-party components may stop supporting outdated versions of their products: in this case, there will be no security updates for new vulnerabilities.
- ICS software is tested and certified to work on certain hardware tools in a given program environment, and depending on the validity conditions of the system certificate, the application of a patch can lead

to a costly and lengthy re-certification procedure for the system.

An important factor is confidence in the source of updates and software developers. As shown by practice, an update may contain deliberately embedded vulnerabilities [2, 34, 35]. Moreover, confidence in the source of updates can change over time, from widespread use of programs to their prohibition, only based on risk assessments related to the social and political circumstances [36, 37].

Let us discuss the testing of updates. Installation and testing of updates may require testing of the entire ICS. In most cases, a hybrid digital twin is a solution to perform full-fledged tests comparable to tests on the real object. In such a twin, some elements of the real ICS equipment are used together with purely digital components [38].

Thus, the practice of sequentially analyzing separate vulnerabilities in components and patching those components can be used on simple systems only. For large and complex systems with a long operation period, it is necessary to find other solutions of the cybersecurity problem.

A list of problems described in the literature was compiled in the review [17]. As we believe, the following are the most important ones:

– The problems of applying patches to ensure information security and current tasks (e.g., continuous business or industrial processes) may be incompatible (see [17], *item 2* of *Table 5*).

– The process of applying patches requires additional resources and expert knowledge, which are not available in the companies operating the software (see [17], *items 5* and *6* of *Table 5*).

– The automatic testing of patches is extremely difficult, most patches are tested manually and the quality of testing is often unsatisfactory (see [17], *items 11* and *12* of *Table 5*).

– Verifying the correctness of an applied patch and eliminating the consequences of deployment errors are a rather difficult problem (see [17], *items 13* and *14* of *Table 5*).

Below we describe a novel risk-oriented approach to simplify vulnerability analysis significantly.

3. THE RISK-ORIENTED APPROACH TO VULNERABILITY ANALYSIS CONSIDERING PROTECTION MEASURES

The approach proposed here is intended for systems with the following characteristics:

– The software environment of a system is treated as a set of separate “black-box” components interacting with each other through a known set of interfaces.

– A system is created using the architectural principles of encapsulation and domain partitioning [39].

We will consider a system in terms of performing definite functions and examine the contribution of each component to the functions.

This approach proceeds from the following main idea: the vulnerability analysis of system software should answer the question of how dangerous a vulnerability is to a particular system function rather than how critical it is to a particular software component.

The vulnerability analysis method proposed below is based on Failure Mode and Effects Analysis (FMEA), a method for identifying weaknesses in the system architecture to improve its reliability and security [18]. FMEA was developed in the 1940s [40] and initially intended to analyze the reliability or security of technical systems and equipment. Later, modifications of this method were proposed to analyze cybersecurity problems for systems with digital components (System Failure Mode and Effects Analysis, SFMEA) [41, 42]. This paper provides general information about FMEA; the interested reader can find details from the extensive literature. Within FMEA, a system must have the following properties:

– definite targets in the form of requirements for its functions,

– established operation conditions,

– definite bounds,

– a hierarchical structure.

Together with the block diagram of the hierarchical structure of elements, FMEA uses block diagrams reflecting the hierarchy of functions performed by system elements and functional relationships between the elements, which makes the functional failures of the system traceable.

The following items are analyzed for each system component (Fig. 1):

– the causes of a given failure;

– a function or failure that can occur (the type of failure);

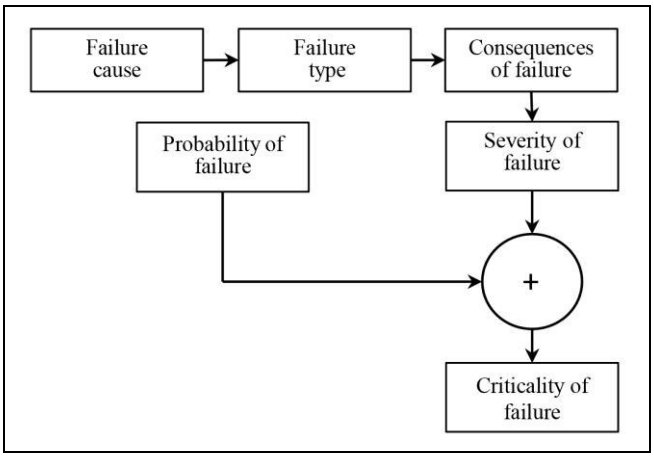

Fig. 1. The flowchart of failure criticality assessment according to the standard [18].

– the peculiarities of possible consequences in case of a failure;

– the severity of a failure (whether the failure is harmless or causes damage);

– the criticality of a failure (how and when the failure can be detected).

When applying FMEA approaches to cybersecurity assessment, it is necessary to describe FMEA stages in cybersecurity terms and relate the cause of a failure to the vulnerability and the intruder's ability to exploit it.

We propose adapting the FMEA methodology in order to assess the impact of a vulnerability, thereby reducing the amount of information under analysis (Fig. 2).

The methodology takes into account that a vulnerability in itself is not the cause of a failure. That is, a vulnerability leads to a failure if an intruder with sufficient competence exploits it to realize a definite threat (see block 1 in Fig. 2). The intruder and his/her competence can be typified according to an accepted scale:

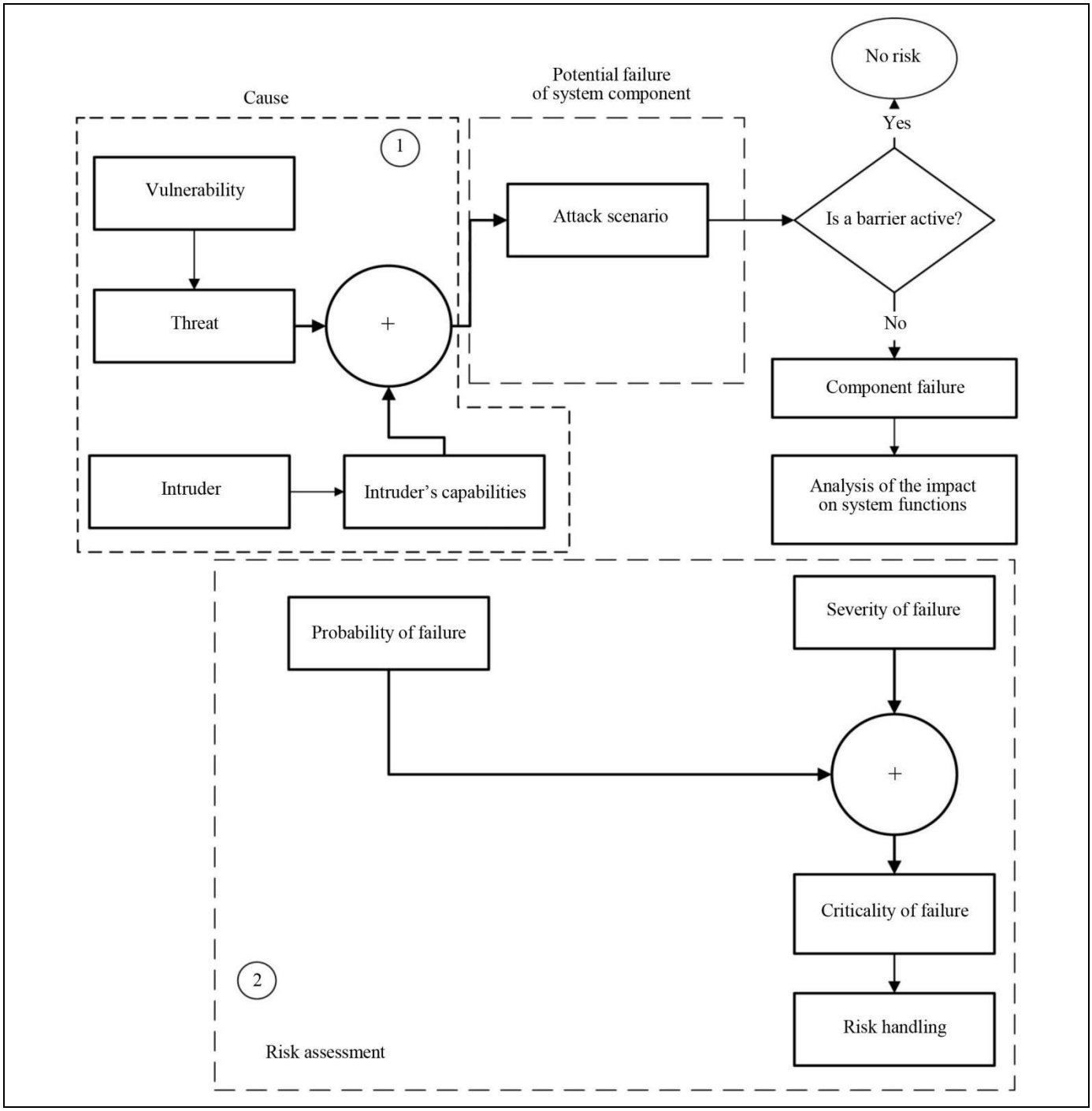

Fig. 2. The flowchart of failure criticality assessment with protection barriers for reliability and cybersecurity problems.

e.g., using the FSTEC classifier [12]. Thus, a failure is the result of a successful attack on the system by an intruder with sufficient competence and capabilities to exploit a vulnerability.

In the methodology proposed, the independence of vulnerabilities is postulated to reduce the analysis space. This is analogous to the assumption on the independence of failures in FMEA ([18], *section 4.1*).

In contrast to the typical consideration of separate technologies underlying a vulnerability (e.g., as in the CWE catalog), the analysis scheme proposed focuses on CVSS attributes to correlate the vulnerability and the associated threat. Additionally, we pass from analyzing separate threats and attackers to analyzing typical attack scenarios associated with threat classes and typical attacker capabilities, thereby reducing the number of threats and attack scenarios under analysis. For ICSs of NPPs, some examples of typical attack scenarios were described in [43].

In this case, a function failure is related not to a particular vulnerability (see Fig. 2) but to the set of conditions and factors that have led to the failure of a component and the intruder's penetration through the protection barrier. Note that a failure is not necessarily directly related to a (cyber)attack and vulnerability exploitation but, e.g., may be the result of resource exhaustion. However, the operation modes of critical facilities are designed so that to exclude resource exhaustion.

The second important supplement to the application of FMEA approaches to cybersecurity problems is to consider the presence of a barrier reflecting the effect of an already implemented set of protection measures. A barrier can block the impact of a component failure on the function of the entire system, thereby nullifying the risks associated with an attack on the system. By assumption, a barrier has a high degree of confidence and can counteract the exploitation of one or more types of vulnerabilities. Thus, when analyzing attack scenarios, a barrier is supposed to be absolute. This model assumption allows significantly reducing the analysis space.

The presence of an intruder with motivation and purpose means that a cybersecurity failure is generally not a random event. Therefore, as a rule, statistical approaches are inapplicable to failure analysis whereas a risk-oriented approach and logical rules can be used.

If the probabilistic nature of the intruder's impact on a system is allowed, the impact of a failure on the entire system and the associated risks (see block 2 in Fig. 2) are assessed using FMEA and the theoretical and probabilistic approaches developed in [18, 44].

The approach described above can be combined into an integrated protection and vulnerability assessment method, called Vulnerability Inspection Control Strategy (VICS); see the flowchart in Fig. 3.

Consider the sequence of actions to analyze vulnerabilities and assess patches for a modern ICS [45] within VICS. The initial data for the analysis are:

– the list of vulnerabilities analyzed;

– an accepted classification system for threats acting on the system (e.g., the CIA model or the FSTEC classifier);

– an accepted intruder's model;

– accepted typical attack scenarios, which allow determining the potential type of a component failure for each "threat class–intruder competence" pair;

– the list of system functions, including known negative consequences of violating them;

– a "function failure–barrier" table, which describes a bigraph defining a barrier for each failure type according to the protection measures implemented;

– the structural diagram of the system, which serves to relate a vulnerability to one or more system components;

– a fault tree, which is intended to trace the impact of a failure of compromised components on system functions and group them by type (see *Fault Tree Analysis* (FTA)).

The analysis comes to the following actions:

1. For each vulnerability, assign a threat class, a typical attack scenario, and a component failure type, considering the intruder's competence level and motivation.

2. Analyze the availability of a protection barrier. If the barrier exists, stop the analysis for this vulnerability.

3. If the barrier is absent or insufficient for this type of attack, select a protection measure or apply a patch that neutralizes the security threats identified or mitigates the risk of their exploitation to an acceptable level, following an accepted patch management methodology (e.g., see the guidelines [9]). If the decision is to implement a new protection measure, add the corresponding attack scenario and barrier into the table describing the bigraph.

Thus, a security update is applied as a vulnerability risk mitigation measure under the following conditions (criteria):

– no barrier against a given attack type;

– an insufficient barrier against a given type of attack;

– the absence of protection measures that are more effective than installing an update.

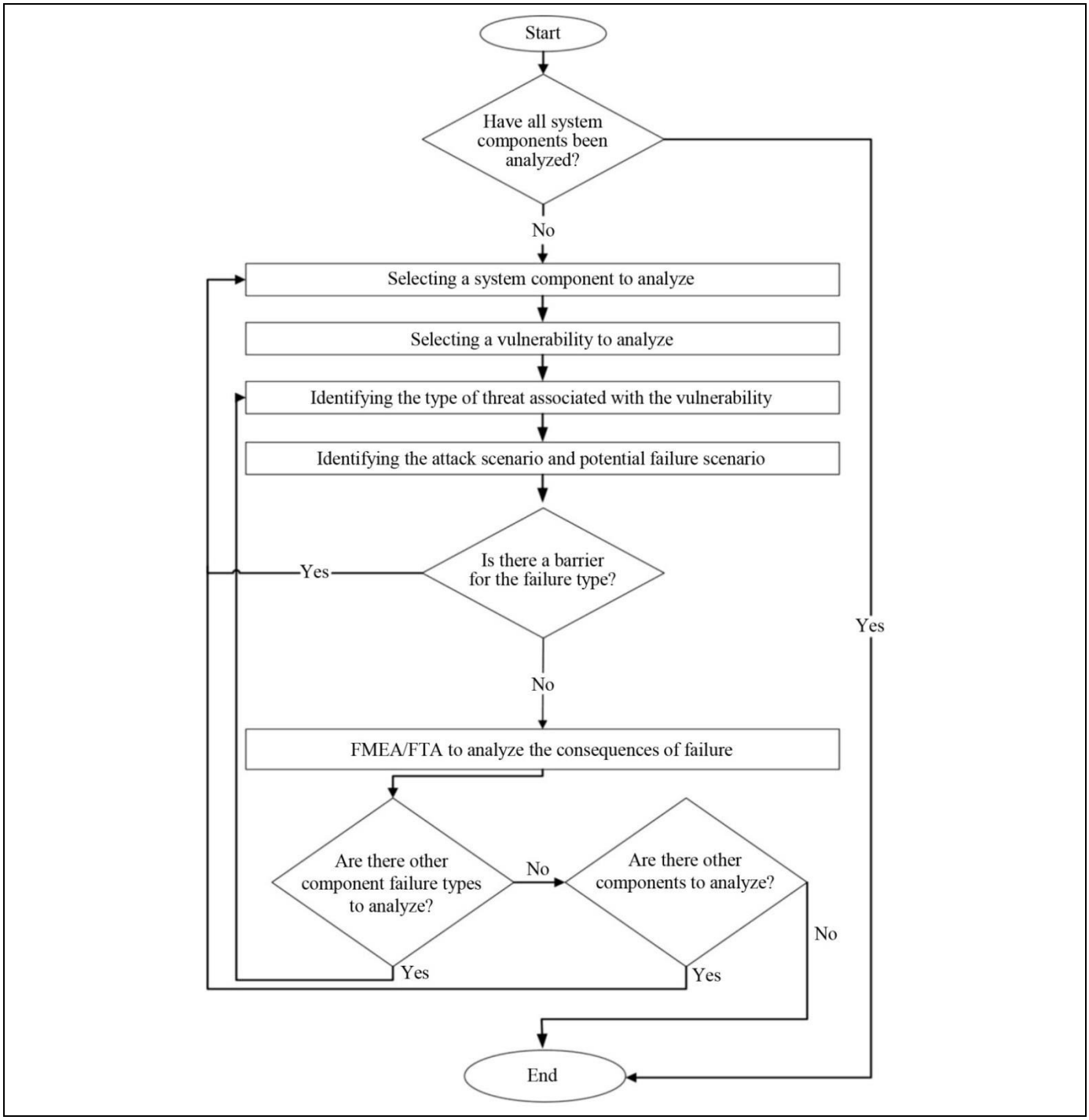

Fig. 3. The flowchart of Vulnerability and Inspection Control Strategy.

We propose the following sequence of actions for information system vulnerabilities:

1. During the development of a protection system, perform a threat analysis and create a catalog of protection barriers and a catalog of typical attack scenarios.

2. Based on the two catalogs, form a bigraph describing the parrying of attacks by protection barriers.

3. Classify each new vulnerability in accordance with the classes of threats, intruders, and attack scenarios (see Section 4). If a vulnerability is assigned a class with an available barrier (a set of protection measures), no patch is required.

4. If a vulnerability leads to a new attack scenario, either install a patch or implement other risk mitigation measures. In particular, they may include the introduction of new protection barriers minimizing the impact of this attack scenario.

5. Analyze periodically the correctness of the protection measures that form barriers. If protection

measures are considered insufficient based on the analysis results, other or additional ones (including patch installation) must be developed and implemented.

Note that VICS does not cover vulnerabilities in software protection barriers and the correctness of the operating environment. However, the systems under consideration are usually built with high confidence in the correctness of protection measures implemented and the quality of system operation; as a result, the number of vulnerabilities in the barriers is usually much smaller than the total number of vulnerabilities in the system. Therefore, the method covers the vast majority of vulnerabilities. Existing guidelines can be used to manage vulnerabilities in protection barriers (e.g., see [3, 8, 9]).

In the next section, we demonstrate the practical application of VICS on the example of building a protection system for an upper-level system of an instrumentation and control system (ULS ICS).

4. PRACTICAL APPLICATION OF THE METHOD

Let us demonstrate the practical application of VICS on the example of ULS ICS. Here is a brief description of its main functions and properties. (For a detailed consideration of such systems, we refer, e.g., to [45].)

The upper-level system of an ICS is intended:

– to implement information, control, and auxiliary functions;

– to send operator's control commands for industrial processes and equipment;

– to monitor the ICS state;

– to integrate information from ICS subsystems.

The ULS ICS under consideration consists of servers, which collect and archive information from related systems and operator's commands, and workstations, where information about the ICS state is displayed and control commands are entered. All ULS elements are redundant and connected by a redundant network. The ICS has no access to the Internet. Unidirectional data flows from the ICS to the outside (e.g., via a data diode) are allowed.

We adopt the CIA model to describe threats, associating with it the main threat types of violating confidentiality, integrity, and availability. This approach is undoubtedly a simplification, but it follows the practice reflected in many widespread security classifiers (e.g., CWE and CVSS). We take the intruder's model with a low or medium level of privileges (no administrator rights) and medium capabilities to realize information security threats (intruder's competence level H3 according to FSTEC). Suppose that the intruder undertakes a local attack to violate the integrity of software or data (including an unauthorized execution of commands) and implement thereby a control command that will cause physical damage to the industrial facility.

Let a set of protection measures (a barrier) be included in the ICS during the design stage to prevent access of an unprivileged user with a medium level of competence in the system software. For the ICS, the protection measures can be selected from the list [46]. Assume also that the system is not compromised at the time of the attack. For the attack scenario and barrier, a fragment of the bigraph is shown in Fig. 4.

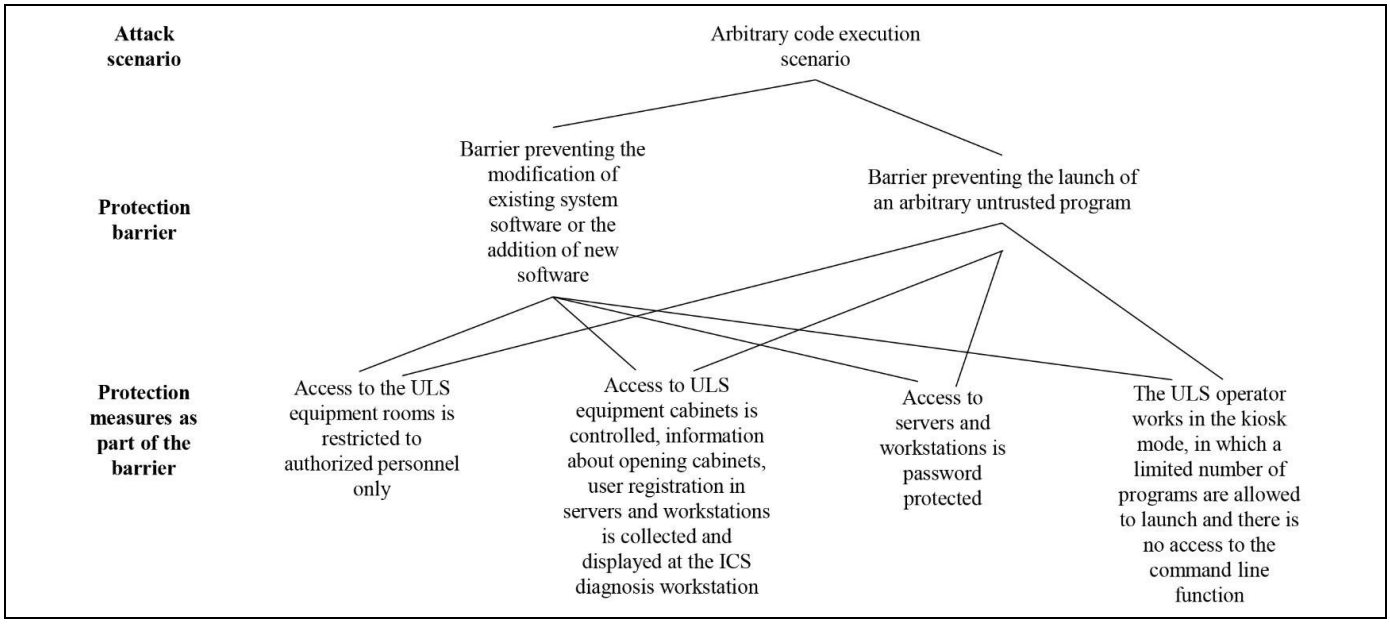

Fig. 4. The bigraph describing the "attack scenario–protection barrier" relationship.

Now we pass to the vulnerability analysis. From the vulnerability database it is necessary to unload the descriptions of vulnerabilities related to the software used in the ULS ICS, considering the versions of all components. To ease the work, one can use a vulnerability scanner.

For the sake of definiteness, let the ULS ICS be implemented in Linux. Consider a vulnerability in the glibc base component (CVE-2020-1752), which has a high degree of danger according to the CVSS 3.0 classifier. Exploiting this vulnerability, a local intruder can execute an arbitrary code by passing a special file path to a program and thus violate software integrity. All these properties of the vulnerability are reflected in the CVSS 3.0 vector.

In the accepted security model (see the table), such a vulnerability corresponds to an attack scenario allowing the execution of an arbitrary code without privilege escalation; also, a barrier acts against this attack to prevent vulnerability exploitation by an intruder in the given attack scenario.

Thus, in view of high confidence in the barrier's capability to counteract uncontrolled access to system software, we consider the risk of vulnerability minimal and no patches need to be installed for the CVE-2020-1752 vulnerability.

Obviously, for vulnerabilities with a similar attack scenario, VICS will yield the same results.

CONCLUSIONS

The problems of installing security updates are quite acute due to the increasing focus on cybersecurity in all computer applications. Recommendations on patching have been developed over the years [3, 8, 9], but their use encounters difficulties in practice.

The first difficulty, which concerns information systems of any purpose, is the effect of scale. Currently, thousands of vulnerabilities are discovered every year, making it inefficient to analyze each vulnerability "manually." The difficulties of analyzing each vulnerability separately can lead to a "patch everything" strategy. However, this strategy is fraught with compatibility validation problems for software and hardware components and disruption of continuous business processes under protection.

Instrumentation and control systems (ICSs) are additionally characterized by a long lifecycle (decades), low variability of hardware during operation, a rather lengthy development stage, and (often) the need for software certification. Therefore, even if all known vulnerabilities are eliminated in the ICS software at the time of its acceptance for operation, by the time of complete commissioning the software will certainly contain newly discovered vulnerabilities that cannot be promptly eliminated. Due to invariable hardware, after several years of operation, many software components will be impossible to update without violating the hardware-software compatibility of the system, as new versions of third-party software may no longer support obsolete hardware. Industry regulations may require recertification after updates have been installed, resulting in additional time and costs. The need for recertification after each software change is essential and is recognized not only by developers but also by regulating authorities. In particular, at the international level, the possibility of classifying changes (patches) according to the degree of their impact on the functionality of software and, depending on this, changing the requirements for recertification is being considered.

Obviously, some method is needed to analyze a large number of vulnerabilities for a particular system and provide recommendations on patching without a full and detailed analysis of each vulnerability.

This paper has proposed an FMEA-based approach considering not a separate vulnerability but its impact as part of an attack scenario (within an accepted model of threats and intruder) on the functions of the entire system with existing barriers (sets of protection measures effective against a definite attack scenario).

In accordance with secure design principles, definite sets of protection measures are embedded into the system to form guaranteed barriers for definite classes of vulnerabilities. These barriers are activated during the operation stage.

The "attack scenario– barrier" relationships form a bigraph, and risk analysis in case of new vulnerabilities is reduced to analyzing this bigraph.

Suppose that a new vulnerability is discovered, leading to an attack scenario without an installed (or ineffective) protection barrier. In this case, the new risk source requires additional measures in the form of installing a security update or applying other compensating measures. This is the decision criterion for installing security updates.

Barriers are considered in the context of protection against classes of vulnerabilities rather than a separate vulnerability. Therefore, the approach not only counteracts open (known) vulnerabilities but also provides protection against yet-to-be-discovered vulnerabilities, which are always present in complex software products.

Note finally that in a complex system, manual vulnerability analysis, even with a small number of attack scenarios, types of intruders and threats, is extremely difficult and makes a mass problem. As we believe, automation of vulnerability analysis using formal

problem-oriented languages for describing vulnerability and attack scenarios will make this approach applicable to real control systems, and research in this direction is ongoing.

REFERENCES

1. *IEC TR 62443-2-3. Technical Report. Security for Industrial Automation and Control Systems. Part 2-3: Patch Management in the IACS Environment*, Geneva: International Electrotechnical Commission, 2015.
2. *The Methodology for Testing Security Updates of Software and Programmable Digital Items*. Approved by the Federal Service for Technical and Export Control (FSTEC) of Russia on October 28, 2022. (In Russian.)
3. Souppaya, M. and Scarfone, K., Guide to Enterprise Patch Management Planning: Preventive Maintenance for Technology, *Special Publication (NIST SP) no. 800-40r4*, Gaithersburg, MD: National Institute of Standards and Technology, 2022. DOI: 10.6028/NIST.SP.800-40r4
4. National Information Assurance (IA) Glossary, *CNSS Instruction no. 4009*, Fort Meade: Committee on National Security Systems Instruction, 2015.
5. *CVE*. URL: https://cve.mitre.org. (Accessed October 2, 2024.)
6. *The Data Bank of Information Security Threats*. URL: https://bdu.fstec.ru/vul. (Accessed October 2, 2024.) (In Russian.)
7. *CERT-FR avis*. URL: https://www.cert.ssi.gouv.fr/avis/. (Accessed February 2, 2024.)
8. *ISO/IEC TS 9569:2023. Technical Specification. Information Security, Cybersecurity and Privacy Protection – Evaluation Criteria for IT Security – Patch Management Extension for the ISO/IEC 15408 series and ISO/IEC 18045*, Geneva: International Standard Organization/International Electrotechnical Commission, 2023.
9. *Vulnerability Management Guidelines for an Authority (Organization)*. Approved by the Federal Service for Technical and Export Control (FSTEC) of Russia on May 17, 2023. (In Russian.)
10. *The Methodology for Assessing the Criticality of Vulnerabilities in Software and Programmable Digital Items*. Approved by the Federal Service for Technical and Export Control (FSTEC) of Russia on October 28, 2022. (In Russian.)
11. Scarfone, K., Souppaya, M., and Dodson, D., Secure Software Development Framework (SSDF). Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities, *Special Publication (NIST SP) no. 800-218*, Gaithersburg, MD: National Institute of Standards and Technology, 2022. DOI: 10.6028/NIST.SP.800-218
12. *The Methodology for Assessing Information Security Threats*. Approved by the Federal Service for Technical and Export Control (FSTEC) of Russia on February 5, 2021. (In Russian.)
13. *IEC/TS 62443-1-1:2009. Industrial Communication Networks. – Network and System Security. – Part 1-1: Terminology, Concepts and Models (IDT)*, Geneva: International Electrotechnical Commission, 2009.
14. *GOST* (State Standard) *R 56939-2016: Information Security. Development of Secure Software. General Requirements*, Moscow: Standartinform, 2016.
15. *GOST* (State Standard): *The Unified System of Program Documentation. The Types of Programs and Program Documents*, Moscow: Standartinform, 2010. (In Russian.)
16. Promyslov, V.G. and Zharko, E.F., Approaches to Risk Assessment in Cybersecurity of A-plant Process Control Systems, *Automation in Industry*, 2022, no. 11, pp. 28–33. (In Russian.)
17. Dissanayake, N., Jayatilaka, A., Zahedi, M., and AliBabar, M., Software Security Patch Management - A Systematic Literature Review of Challenges, Approaches, Tools and Practices, *Information and Software Technology*, 2022, vol. 144, art. no. 106771.
18. *IEC 60812:2006. Analysis Techniques for System Reliability – Procedure for Failure Mode and Effects Analysis (FMEA)*, Geneva: International Electrotechnical Commission, 2006.
19. *About CWE*. URL: https://cwe.mitre.org/about/index.html. (Accessed April 22, 2024.)
20. *GOST* (State Standard) *R ISO/MEK 27005-2010*: *Information Technology. Methods and Means of Ensuring Security. Risk Management in Information Security*. Moscow: Standartinform, 2011. (In Russian.)
21. *GOST* (State Standard) *R ISO/MEK 15408-3-2008: Information Technology. Methods and Means of Ensuring Security. Criteria for Assessing the Security of Information Technology. Part 3. Security Confidence Components*, Moscow: Standartinform, 2008. (In Russian.)
22. *CVE Metrics*. URL: https://www.cve.org/About/Metrics. (Accessed April 22, 2024.)
23. Haddad, A., Aaraj, N., Nakov P., et al., Automated Mapping of CVE Vulnerability Records to MITRE CWE Weaknesses, *arXiv:2304.11130*, 2023. DOI: 10.48550/arXiv.2304.11130
24. Lin, Y.-Z., Mamun, M., Chowdhury, V.A., et al., HW-V2W-Map: Hardware Vulnerability to Weakness Mapping Framework for Root Cause Analysis with GPT-assisted Mitigation Suggestion, *arXiv:2312.13530*, 2023. DOI: 10.48550/arXiv.2312.13530
25. Mell, P., Scarfone, K., and Romanosky, S., A Complete Guide to the Common Vulnerability Scoring System Version 2.0, 2007. URL: https://www.first.org/cvss/v2/guide. (Accessed April 1, 2024.)
26. *Common Vulnerability Scoring System v3.0: Specification Document*, URL: https://www.first.org/cvss/v3.0/specification-document. (Accessed September 22, 2024.)
27. *CVSS V3 Calculator*. URL: https://bdu.fstec.ru/calc3. (Accessed September 22, 2024). (In Russian.)
28. Kekül, H., Ergen, B., and Arslan, H., Comparison and Analysis of Software Vulnerability Databases, *Int. J. Eng. Manuf.*, 2022, vol. 12, no. 4, pp. 1–14.
29. *How We Assess Acceptance Levels*. National Vulnerability Database. URL: https://nvd.nist.gov/vuln/cvmap/How-We-Assess-Acceptance-Levels. (Accessed April 26, 2024.)
30. *CAPEC List*. URL: https://capec.mitre.org/data/index.html. (Accessed April 22, 2024.)
31. *MITRE ATT&CK*. URL: https://attack.mitre.org/. (Accessed April 22, 2024.)
32. *Microsoft Threat Modeling Tool*. URL: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats. (Accessed May 31, 2024.)
33. Pietre-Cambacedes, L. and Chaudet, C., Disentangling the Relations between Safety and Security, *Proceedings of the 9th WSEAS International Conference on Applied Informatics and Communications*, Stevens Point, Wisconsin, 2009, pp. 156–161.
34. Boehs, E., Everything I Know About the XZ Backdoor. URL: https://boehs.org/node/everything-i-know-about-the-xz-backdoor. (Accessed April 10, 2024.)
35. *CVE-2022-23812*. URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-23812. (Accessed April 10, 2024.)

36. Kapranov, O. and Gureeva, Yu., Employees of the Ministry of Education Have Been Prohibited from Using Apple Devices, *Rossiiskaya Gazeta*, July 7, 2023. URL: https://rg.ru/2023/07/19/sotrudnikam-minprosveshcheniia-zapretili-polzovatsia-tehnikoj-apple.html. (Accessed October 22, 2024; in Russian.)
37. *Commerce Department Prohibits Russian Kaspersky Software for U.S. Customers*, Bureau of Industry and Security, June 20, 2024. URL: https://www.bis.gov/press-release/commerce-department-prohibits-russian-kaspersky-software-us-customers. (Accessed December 20, 2024.)
38. Semenkov, K., Promyslov, V., Poletykin, A., et al., Validation of Complex Control Systems with Heterogeneous Digital Models in Industry 4.0 Framework, *Machines*, 2021, vol. 9, no. 3, art. no. 62.
39. *GOST* (State Standard) *ISO/IEC TS 19249-2021: Information Technologies. Methods and Means of Ensuring Security. The Catalog of Architecture and Design Principles for Secure Products, Systems, and Applications*, Moscow: Standartinform, 2021. (In Russian.)
40. *MIL-P 1629: USA Military Standard, Procedure for Performing a Failure Mode, Effects and Criticality Analysis*, Washington, DC: Department of Defense, 1980.
41. Schmittner, C., Gruber, T., Puschner, P., and Schoitsch, E., Security Application of Failure Mode and Effect Analysis (FMEA), in *Lecture Notes in Computer Science*, 2014, vol. 8666, pp. 310–325. DOI: https://doi.org/10.1007/978-3-319-10506-2_21
42. Talwar, P. Software Failure Mode and Effects Analysis, Advances in Intelligent Systems and Computing, 2020, vol. 1131, pp. 86–91.
43. Busquim e Silva, R.A., Piqueira, J.R.C., Cruz, J.J., Marques R.P. Cybersecurity Assessment Framework for Digital Interface Between Safety and Security at Nuclear Power Plants, International Journal of Critical Infrastructure Protection, 2021, vol. 34, art. no. 100453. DOI: 10.1016/j.ijcip.2021.100453
44. Kalashnikov, A.O., Bugajskij, K.A., Birin, D.S., et al., Application of the Logical-Probabilistic Method in Information Security (Part 1), *Cybersecurity Issues*, 2023, no. 4 (56), pp. 23–32. (In Russian.)
45. Mengazetdinov, N.E., Poletykin, A.G., Promyslov, V.G., et al., *Kompleks rabot po sozdaniyu pervoi upravlyayushchei sistemy verkhnego blochnogo urovnya ASU TP dlya AES "Busher" na osnove otechestvennykh informatsionnykh tekhnologii* (The Complex of Works on Creating the First Control System of the Upper Block Level of the ICS for the Bushehr NPP Based on Russian Information Technology), Moscow: Trapeznikov Institute of Control Sciences RAS, 2013. (In Russian.)
46. *Requirements for Ensuring the Security of Significant Critical Information Infrastructure Objects of the Russian Federation*. Approved by the Federal Service for Technical and Export Control (FSTEC) of Russia on December 25, 2017. (In Russian.)

This paper was recommended for publication by R. V. Meshcheryakov, a member of the Editorial Board.

Received June 6, 2024, and revised March 18, 2025.
Accepted April 28, 2025

Author information

Semenkov, Kirill Valer'evich. Cand. Sci. (Phys.–Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Science, Moscow, Russia
✉ semenkov@ipu.ru
ORCID iD: https://orcid.org/0000-0003-0865-9072

Promyslov, Vitaly Georgievich. Cand. Sci. (Phys.–Math.), Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ vp@ipu.ru
ORCID iD: https://orcid.org/0000-0003-1919-8718

Cite this paper

Semenkov, K.V., and Promyslov, V.G., A Procedure for Assessing Security Updates in Industrial Systems. *Control Sciences* **2,** 50–62 (2025).

Original Russian Text © Semenkov, K.V., Promyslov, V.G., 2025, published in *Problemy Upravleniya*, 2025, no. 2, pp. 58–73.

Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com